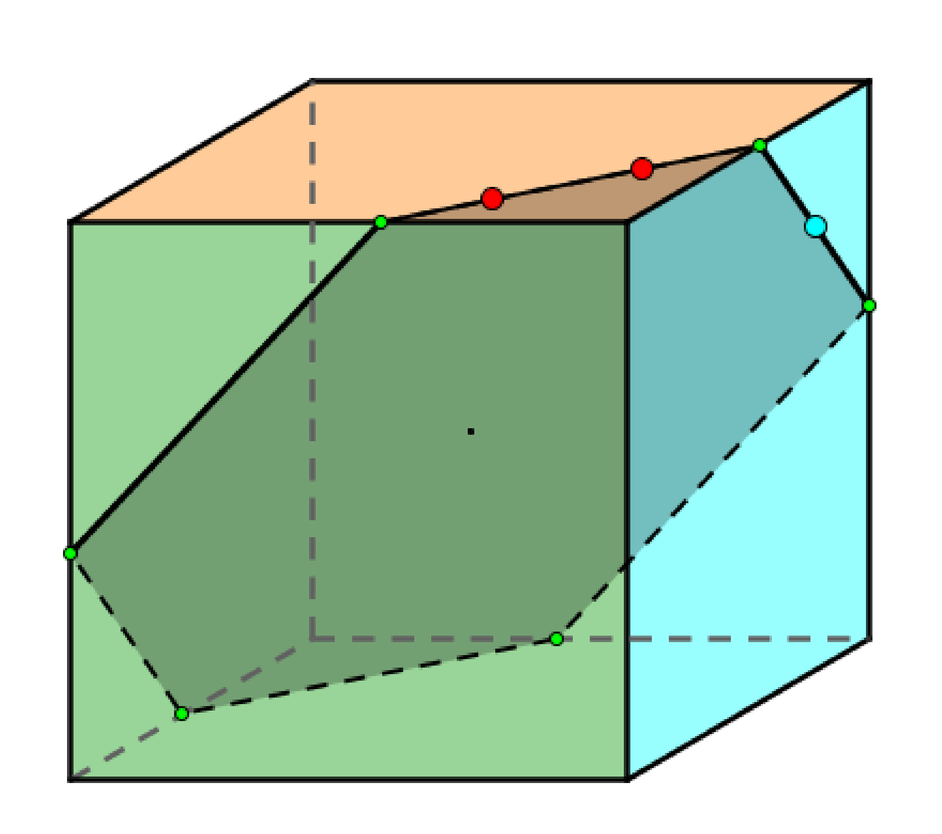


Section d'un plan dans un cube

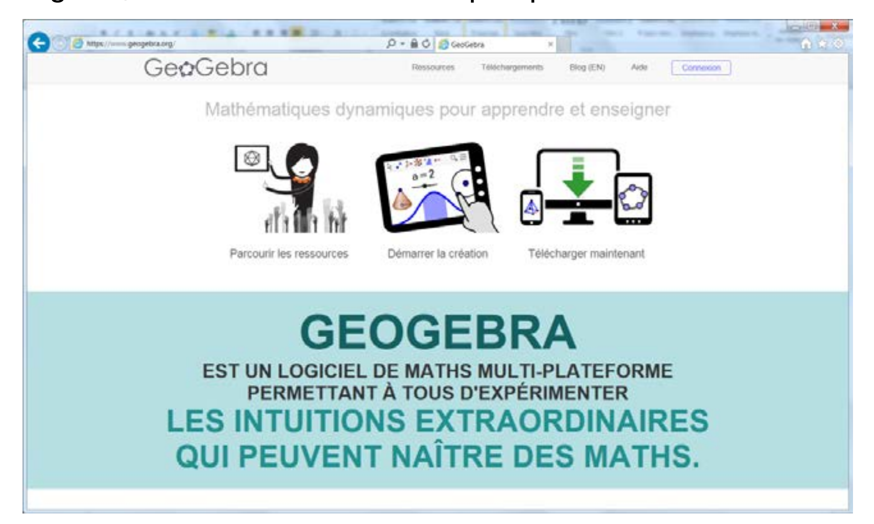
construite

avec

Geogebra

« Sectioncube », une récréation cérébrale « géogébrique » pour vacances pluvieuses.

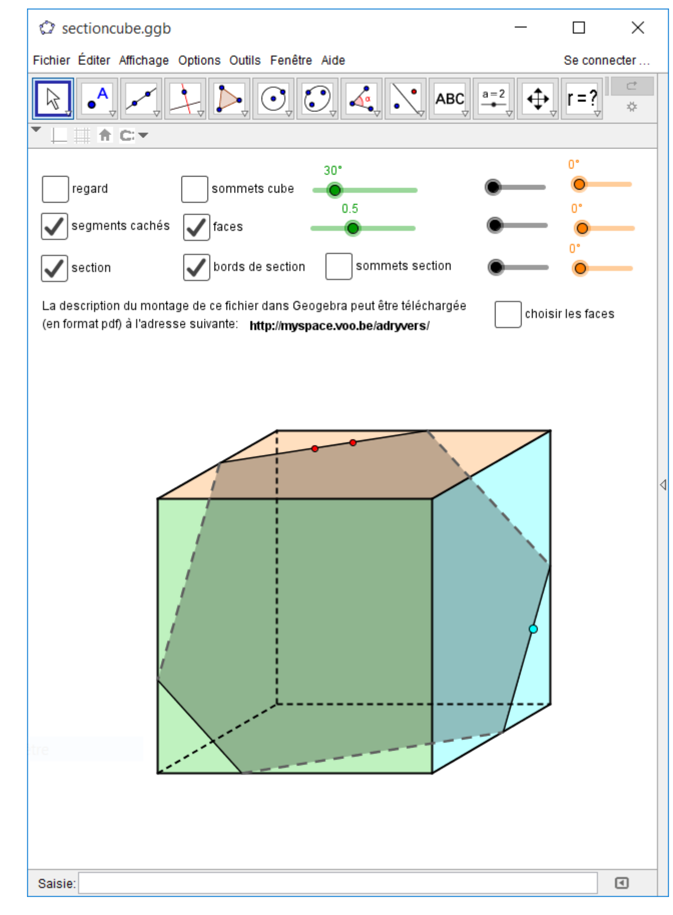
« Sectioncube » est une vue en perspective de la section d'un plan dans un cube, ce plan étant déterminé par deux points situés dans une première face du cube et par un troisième situé dans une seconde face. La construction de ce fichier peut se faire avec le logiciel « Geogebra », téléchargeable ***gratuitement*** sur Internet à l'adresse http://www.geogebra.org. La fabrication de « Sectioncube » (dans Geogebra, par vos soins) vous permettra, en plus de faire connaissance avec ce logiciel, de remettre en œuvre quelques souvenirs mathématiques.



Pour toutes les descriptions de constructions de fichiers ou macros (outils) dans les pages suivantes, **nous nous conformerons aux 2 points de vue suivants**.

- Ces pages ne sont pas un mode d'emploi pour Geogebra. Elles expliquent seulement la construction des vues en perspective de la section d'un plan dans un cube dans Geogebra. Si vous ne connaissez pas Geogebra, il faudra préalablement prendre connaissance de son fonctionnement en vous servant de l'aide proposée par Geogebra,

- Les noms donnés aux objets peuvent paraître longs. Cependant, plus le nombre d'objets augmente (et dans la construction de la section d'un plan dans un cube, il y a beaucoup d'objets géométriques !), plus il deviendrait difficile de gérer les constructions si les noms des objets étaient sans rapport avec les natures de ceux-ci. Les sommets de deux quadrilatères sont plus faciles à retrouver s'ils s'appellent « fS1 », « fS2 », « fS3 » et « fS4 » pour le premier et « mS1 », « mS2 », « mS3 » et « mS4 » pour le second plutôt que, par exemple, « A_1 », « X6_1 », « T », « G_2_2 », « P », « Gh », G_1 » et « G_2 ».



Elaboration du fichier « Sectioncube.ggb »

Nous nous proposons de réaliser un fichier Geogebra présentant une représentation en perspective de la section d'un plan dans un cube, avec ou sans couleurs, avec ou sans les parties cachées, avec ou sans les sommets, tournant dans l'espace sous 3 directions, indépendamment l'une de l'autre ou simultanément. La représentation en perspective sera paramétrable au niveau du point de vue (montant ou descendant), de l'angle de fuite, du coefficient de réduction et des 3 angles de rotation autour des axes du repère de l'espace contrôlant ce cube, la vitesse de chacun de ces angles étant contrôlée par un curseur linéaire.

Le plan sera déterminé par trois points : les deux premiers seront placés dans une des six faces du cube tandis que le troisième sera placé dans une autre des six faces du cube.

Les explications concernant la construction et le paramétrage de la construction d'un hexaèdre vu en perspective cavalière peuvent être trouvées dans le fichier « platon.pdf » téléchargeable sur http://myspace.voo.be/adryvers.

Préparons les objets qui permettront de contrôler les animations de notre fichier.

Ouvrons donc les macros « repère3d.ggt », « indicateurface.ggt » et « hexaèdre.ggt ». Sauvons le tout dans un fichier « sectioncube.ggb » et construisons le contenu de ce fichier.

Nous utiliserons une boîte à cocher « regard » qui déterminera la manière dont nous déciderons d'observer le cube (vue montante ou descendante). Sa légende sera « regard ».

Nous utiliserons une boîte à cocher « caché » qui permettra de masquer ou non les segments cachés dessinés en pointillés. Sa légende sera « segments cachés ».

Nous utiliserons une boîte à cocher « vuesection » qui permettra de masquer ou non la section. Sa légende sera « section ».

Nous utiliserons une boîte à cocher « sommets » qui permettra de masquer ou non les sommets du cube. Sa légende sera « sommets cube ».

Nous utiliserons une boîte à cocher « faces » qui permettra de masquer ou non les couleurs des faces. Sa légende sera « faces ».

Nous utiliserons une boîte à cocher « bordssection » qui permettra de masquer ou non les bords de la section. Sa légende sera « bords de section ».

Nous utiliserons une boîte à cocher « sommetssection » qui permettra de masquer ou non les sommets de la section. Sa légende sera « sommets section ».

Nous utiliserons un curseur angle « fuite » pour paramétrer l'angle de fuite de la perspective cavalière.

Nous utiliserons un curseur nombre « réduction » pour paramétrer le coefficient de réduction de la perspective cavalière.

Nous utiliserons un curseur angle « α » pour faire tourner le repère autour de l'axe des X. Un curseur nombre « vα » l'accompagnera pour contrôler sa vitesse d'animation.

Nous utiliserons un curseur angle « β » pour faire tourner le repère autour de l'axe des Y. Un curseur nombre « vβ » l'accompagnera pour contrôler sa vitesse d'animation.

Nous utiliserons un curseur angle « γ » pour faire tourner le repère autour de l'axe des Z. Un curseur nombre « vγ » l'accompagnera pour contrôler sa vitesse d'animation.

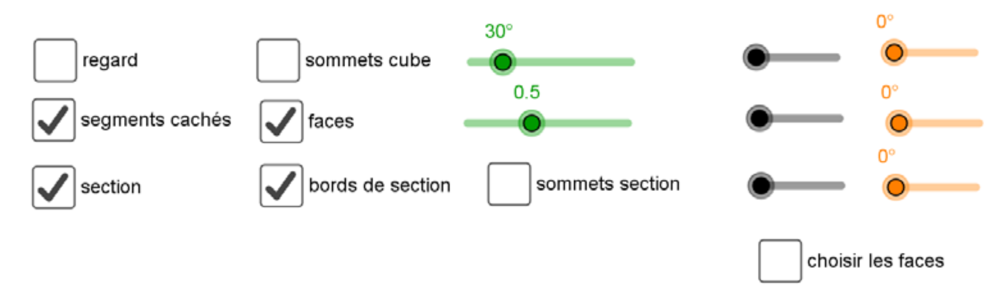
Nous utiliserons une boîte à cocher « ppoints » qui permettra de choisir les deux faces dans lesquelles seront placés les points déterminant la section. Sa légende sera « choisir les faces ».

Nous utiliserons un curseur nombre « tp1 » pour choisir, parmi les six faces du cube celle dans laquelle deux des trois points déterminant le plan pourront circuler et nous utiliserons un

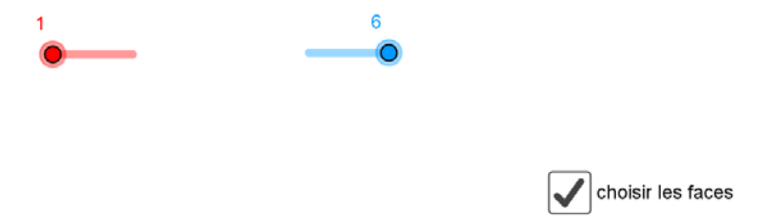
curseur nombre « tp2 » pour choisir, parmi les six faces du cube celle dans laquelle le troisième des points déterminant le plan pourra circuler.

Nous utiliserons enfin un bouton poussoir « reset » pour réinitialiser les valeurs par défaut de α, vα, β, vβ, γ, vγ, fuite, réduction, regard, repère, caché, faces, texte, sommets et points.

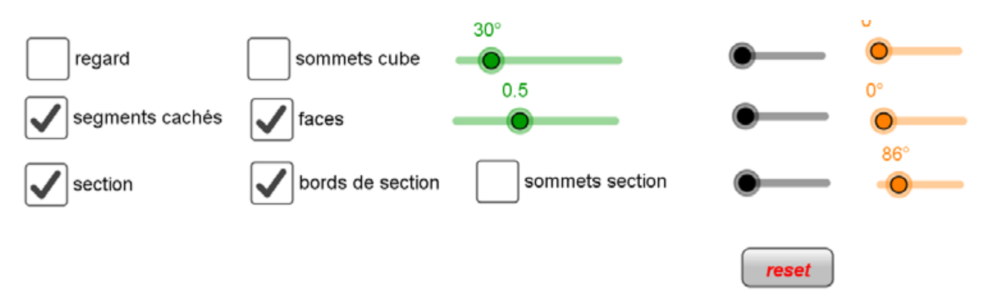
Dans ''Propriétés, onglet Avancé, Condition pour afficher l'objet'', nous écrirons '' ¬ppoints'' pour les objets « regard », « caché », « vuesection », « sommets », « faces », « bordssection », « sommetssection », « fuite », « réduction », « α », « vα », « β », « vβ », « γ » et « vγ ».
Ces objets seront donc visibles lorsque la case « choisir les faces » sera décochée.



Dans ''Propriétés, onglet Avancé, Condition pour afficher l'objet'', nous écrirons '' ppoints'' pour les objets « tp1 » et « tp2 ». Ces objets seront donc visibles seulement lorsque la case « choisir les faces » sera cochée.



Dans ''Propriétés, onglet Avancé, Condition pour afficher l'objet'', nous écrirons $\alpha \stackrel{?}{=} 0° \wedge \beta \stackrel{?}{=} 0° \wedge \gamma \stackrel{?}{=} 0°$ pour l'objet « ppoints » et nous écrirons $\neg(\alpha \stackrel{?}{=} 0° \wedge \beta \stackrel{?}{=} 0° \wedge \gamma \stackrel{?}{=} 0°)$' pour l'objet « reset ». Les bouton « reset » et la case à cocher « choisir les faces » seront donc respectivement invisibles et visibles lorsque les angles « α », « β » et « γ » seront simultanément nuls et seront donc respectivement visibles et invisibles dans le cas contraire. Les raisons de ces choix seront justifiées dans la suite de la construction de la section dans le cube.



1) Curseur : fuite (onglet Intervalle, min : 0°, max : 90°, Incrément : 1°)
2) Curseur : réduction (onglet Intervalle, min : 0, max : 1, Incrément : 0.05)
3) Curseur : vα (onglet Intervalle, min : 0, max : 7, Incrément : 1)
4) Curseur : vβ (onglet Intervalle, min : 0, max : 7, Incrément : 1)
5) Curseur : vγ (onglet Intervalle, min : 0, max : 7, Incrément : 1)
6) Curseur : α (onglet Intervalle, min : 0°, max : 360°, Incrément : 1°, onglet Animation, Vitesse : vα, Répéter, croissant)

7) Curseur : β (onglet Intervalle, min : 0°, max : 360°, Incrément : 1°, onglet Animation, Vitesse : vβ, Répéter, croissant)

8) Curseur : γ (onglet Intervalle, min : 0°, max : 360°, Incrément : 1°, onglet Animation, Vitesse : vγ, Répéter, croissant)

Les curseurs « α », « β » et « γ » seront animés et une simple manipulation des curseurs « vα », «vβ » et « vγ » permettra de contrôler les vitesses de rotation autour des axes, voire même arrêter ces rotations.

9) Booléen : ppoints (légende ‘’choisir les faces’’)

10) Curseur : tp1 (onglet Intervalle, min : 1, max : 6, Incrément : 1, rouge, Condition pour afficher l’objet : ppoints)

11) Curseur : tp2 (onglet Intervalle, min : 1, max : 6, Incrément : 1, bleu, Condition pour afficher l’objet : ppoints)

12) Booléen : regard (légende ‘’regard’’)

13) Booléen : vuesection (légende ‘’section’’)

14) Booléen : caché (légende ‘’segments cachées’’)

15) Booléen : sommets (légende ‘’sommets’’)

16) Booléen : faces (légende ‘’faces’’)

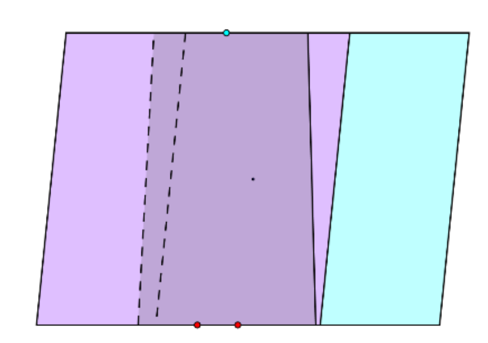
17) Booléen : sommetssection (légende ‘’sommets section’’)

18) Booléen : bordssection (légende ‘’bords section’’)

19) Bouton : reset onglet Script, par Clic, α = 0°, vα = 0, β = 0°, vβ = 0, γ = 0°, vγ = 0, fuite = 30°, réduction = 0.5, regard = false, caché = true, faces = true, sommets = false, vuesection = true, bordssection = true, sommetssection = false, ppoints = false.

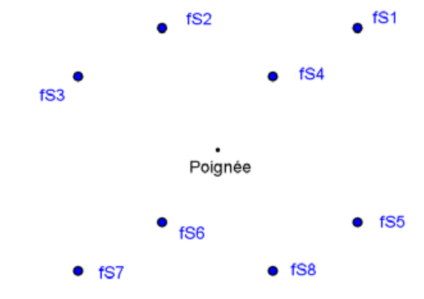
20) Point : Poignée (taille : 1, ne pas afficher l’étiquette)

Nous construirons la section du plan dans un cube fixe. Le faire directement dans un cube mobile ne nous permettrait pas de déterminer les traces du plan dans des faces qui seraient, placées dans des positions particulières, réduites à des segments de droite comme le montre, par exemple, la figure ci-contre. Nous reporterons la section construite sur le cube fixe sur un cube mobile au moyen d’un outil que nous créerons à cet usage et qui sera décrit plus tard.



Appliquons l’outil « repère3d.ggt » sur « Poignée ».

Lorsque Geogebra demandera les paramètres, nous désignerons « fuite », « réduction », « 0° », « 0° » et « 0° ». La macro nous donnera les points déterminant le repère fixe et les vecteurs associés pour lesquels nous choisirons la couleur orange. Nous renommerons ces éléments « fU », « fV », « fW » pour les points et « fu », « fv » et « fw » pour les vecteurs. Nous masquerons les trois vecteurs.



21) Outil repère3d : repère3d[Poignée, fuite, réduction, 0°, 0°, 0°] ; on obtient « fU », « fV », « fW », « fu », « fv » et « fw »



Appliquons l'outil « hexaèdre.ggt » sur «Poignée », « fU », « fV » et « fW ».

Nous renommerons « fS1 », « fS2 », « fS3 », « fS4 », « fS5 », « fS6 », « fS7 » et « fS8 » les sommets ainsi construits en respectant la disposition ci-dessus.
Nous masquerons « fU », « fV » et « fW ».

22) Outil hexaèdre : hexaèdre[Poignée, fU, fV, fW] ;
on obtient « fS1 », « fS2 », « fS3 », « fS4 », « fS5 », « fS6 », « fS7 », et « fS8 »

Plaçons les points fA et fB dans une des faces du cube fixe selon la valeur prise par le curseur nombre « tp1 ». Voici un schéma expliquant la construction de fA.

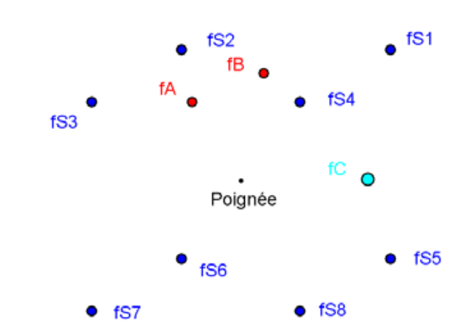
```
Si tp1 = 1
| alors fA est dans le quadrilatère fS1fS2fS3fS4
| sinon  Si tp1 = 2
|        |  alors fA est dans le quadrilatère fS1fS4fS8fS5
|        |  sinon  Si tp1 = 3
|        |         |  alors fA est dans le quadrilatère fS1S2fS6fS5
|        |         |  sinon  tp1 = 4
|        |         |         | alors fA est dans le quadrilatère fS2fS3fS7fS6
|        |         |         | sinon  tp1 = 5
|        |         |         |        |  alors fA est dans le quadrilatère fS4fS3fS7fS8
|        |         |         |        |  sinon fA est dans le quadrilatère fS5fS6fS7fS8
|        |         |         |        Fin alors sinon
|        |         |         Fin si alors sinon
|        |         Fin si alors sinon
|        Fin si alors sinon
Fin si alors sinon
```

23) Saisie : fA = PointDans[Si[tp1 ≟ 1, Polygone[fS4, fS1, fS2, fS3], Si[tp1 ≟ 2, Polygone[fS4, fS1, fS5, fS8], Si[tp1 ≟ 3, Polygone[fS1, fS2, fS6, fS5], Si[tp1 ≟ 4, Polygone[fS2, fS3, fS7, fS6], Si[tp1 ≟ 5, Polygone[fS4, fS3, fS7, fS8], Polygone[fS5, fS6, fS7, fS8]]]]]]
(taille : 4, rouge)

24) Saisie : fB = PointDans[Si[tp1 ≟ 1, Polygone[fS4, fS1, fS2, fS3], Si[tp1 ≟ 2, Polygone[fS4, fS1, fS5, fS8], Si[tp1 ≟ 3, Polygone[fS1, fS2, fS6, fS5], Si[tp1 ≟ 4, Polygone[fS2, fS3, fS7, fS6], Si[tp1 ≟ 5, Polygone[fS4, fS3, fS7, fS8], Polygone[fS5, fS6, fS7, fS8]]]]]] (taille : 4, rouge)

Plaçons le point « fC » dans une des faces du cube fixe selon la valeur prise par « tp2 ».

25) Saisie : fC = PointDans[Si[tp2 ≟ 1, Polygone[fS4, fS1, fS2, fS3], Si[tp2 ≟ 2, Polygone[fS4, fS1, fS5, fS8], Si[tp2 ≟ 3, Polygone[fS1, fS2, fS6, fS5], Si[tp2 ≟ 4, Polygone[fS2, fS3, fS7, fS6], Si[tp2 ≟ 5, Polygone[fS4, fS3, fS7, fS8], Polygone[fS5, fS6, fS7, fS8]]]]]]
(taille : 4, bleu)



Construisons maintenant un cube mobile.

Appliquons l'outil « repère3d.ggt » sur « Poignée ».

Lorsque Geogebra demandera les paramètres, nous désignerons « fuite », « réduction », « α », « β » et « γ ». La macro nous donnera les points déterminant le repère fixe et les vecteurs associés pour lesquels nous choisirons la couleur orange. Nous renommerons ces éléments « mU », « mV », « mW » pour les points et « mu », « mv » et « mw » pour les vecteurs. Nous masquerons les trois vecteurs.

26) Outil repère3d : repère3d[Poignée, fuite, réduction, α, β, γ] ; on obtient « mU », « mV », « mW », « mu », « mv » et « mw »

Appliquons l'outil « hexaèdre.ggt » sur «Poignée », « mU », « mV » et « mW ».

Nous renommerons « mS1 », « mS2 », « mS3 », « mS4 », « mS5 », « mS6 », « mS7 » et « mS8 » les sommets ainsi construits en respectant la disposition ci-contre.
Nous masquerons « mU », « mV » et « mW ».

27) Outil hexaèdre : hexaèdre[Poignée, mU, mV, mW] ;
on obtient « mS1 », « mS2 », « mS3 », « mS4 », « mS5 », « mS6 », « mS7 », et « mS8 »

Construisons les faces et leurs indicateurs de visibilité.

Avec α = 0°, β = 0°, γ = 0°, nous déciderons de nommer « Fa1 », « Fa2 », « Fa3 », « Fa4 », « Fa5 » et « Fa6 » les faces supérieure, latérale droite, arrière, latérale gauche, avant et inférieure.

28) Polygone : Fa1 = Polygone[mS1, mS2, mS3, mS4] (couleur : orange, opacité : 25)

La case « regard » étant décochée, décidons d'adopter le regard de perspective qui rend Fa1 vue. Appliquons donc l'outil « indicateurdeface » sur « mS2 », « mS1 » et « mS4 » (pour respecter le sens horaire et obtenir une valeur « true »).

29) Indicateur de face : indFa1 = indicateurdeface[mS2, mS1, mS4]

30) Polygone : Fa2 = Polygone[mS4, mS8, mS5, mS1] (couleur : bleu, opacité : 25)

La case « regard » étant décochée, décidons d'adopter le regard de perspective qui rend la face 2 vue. Appliquons donc l'outil « indicateurface » sur « mS1 », « mS5 » et « mS8 » (pour respecter le sens horaire et obtenir une valeur « true »).

31) Indicateur de face : indFa2 = indicateurdeface[mS1, mS5, mS8]

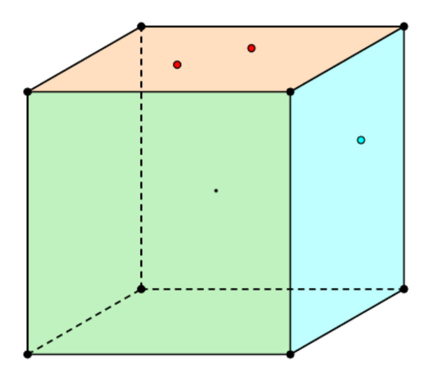
32) Polygone : Fa3 = Polygone[mS1, mS2, mS6, mS5] (couleur : violet, opacité : 25)

La case « regard » étant décochée, décidons d'adopter le regard de perspective qui rend la face 3 cachée. Appliquons donc l'outil « indicateurface » sur « mS1 », « mS2 » et « mS6 » (pour respecter le sens anti horaire et obtenir une valeur « false »).

33) Indicateur de face : indFa3 = indicateurdeface[mS1, mS2, mS6]

34) Polygone : Fa4 = Polygone[mS2, mS3, mS7, mS6] (couleur : rouge, opacité : 25)

Les indicateurs de deux faces opposées sont chacun le contraire de l'autre car dans ce cas, lorsqu'une face est vue, l'autre est cachée. Nous nommerons « indFa4 » le booléen contraire au booléen indFa2.



35) Indicateur de face : indFa4 = ¬indFa2

36) Polygone : Fa5 = Polygone[mS3, mS4, mS8, mS7]
(couleur : vert, opacité : 25)

Fa5 est opposée à Fa3. On aura donc

37) Indicateur de face : indFa5 = ¬indFa3

38) Polygone : Fa6 = Polygone[mS8, mS5, mS6, mS7] (couleur : gris, opacité : 25)

Fa6 est opposée à Fa1. On aura donc

39) Indicateur de face : indFa6 = ¬indFa1

Paramétrons les faces.

Voici un schéma expliquant l'influence des valeurs des cases à cocher « faces » et « regard » sur le caractère visible ou caché de Fa1. Le rôle de la case « regard » est d'inverser (remplacer true par false et inversement) les valeurs des indicateurs de visibilité des faces.

Si la case « faces » n'est pas cochée
| alors Fa1 doit être invisible
| Sinon si la case « regard » n'est pas cochée
| | Alors Fa1 est visible ou invisible selon la valeur « true » ou « False » de indFa1
| | Sinon Fa1 est invisible ou visible selon la valeur « true » ou « False » de indFa1
| Fin si alors sinon
Fin si alors sinon

Dans la fenêtre Algèbre, sélectionnons les faces que nous venons de créer (avec les touches « control » et « majuscules » pour faire des sélections multiples), donc « Fa1 », « Fa2 », « Fa3 », « Fa4 », « Fa5 » et « Fa6 ». Puis, dans Propriétés, onglet Avancé, conditions pour afficher l'objet, écrivons la formule suivante : Si[¬faces, false, Si[¬regard, indFa1, ¬indFa1]].

Grace à notre sélection multiple, cette formule, qui n'est valable que pour « Fa1 » a été écrite pour les six faces. Il sera en effet plus facile de changer quelques chiffres dans les paramètres de chaque face dans une formule (mal écrite mais) déjà écrite que de retaper tous les caractères de cette formule.

Renommons les segments.

La construction des polygones a fait apparaître 24 segments, qui sont bien sûr, les côtés des polygones. Les noms attribués par Geogebra sont tellement diversifiés que nous allons renommer tout cela pour retrouver et gérer ces éléments moins difficilement. Pour nos 12 arrêtes de l'hexaèdre, Geogebra a créé 24 segments puisque les 6 faces carrées ont bien, ensembles, 12 côtés. A chaque arête du polyèdre, nous associerons 2 segments : un en traits pleins pour représenter l'arête lorsqu'elle est vue et un en traits pointillés pour représenter l'arête lorsqu'elle est cachée. Nous renommerons donc les segments de nos 6 carrés comme décrit ci-dessous.

Si nous posons la souris sur le segment du tétraèdre commun aux faces « F1 » et « F2 », nous verrons que deux segments seront sélectionnés dans la fenêtre Algèbre. On en renommera un en « aplF1F2 » et l'autre en « aptF1F2 » ce qui signifie, on le devine ''arête plein face1 face2'' et ''arête pointillés face1 face2''. En continuant de la même manière, on obtiendra donc 12 segments pour les arêtes vues et 12 segments pour les arêtes cachées. En utilisant judicieusement la technique des copier/coller (control c et control v), ce travail peut se faire avec des économies de frappes clavier. Nous obtiendrons donc « aplF1F2 », « aplF1F3 », « aplF1F4 », « aplF1F5 », « aplF2F3 », « aplF2F5 », « aplF2F6 », « aplF3F4 », « aplF3F6 », « aplF4F5 », « aplF4F6 » et « aplF5F6 » pour les arêtes vues et « aptF1F2 », « aptF1F3 », « aptF1F4 », « aptF1F5 », « aptF2F3 », « aptF2F5 », « aptF2F6 », « aptF3F4 », « aptF3F6 », « aptF4F5 », « aptF4F6 » et « aptF5F6 » pour les arêtes cachées.

Paramétrons les arêtes cachées.

Voici un schéma expliquant l'influence des valeurs des cases à cocher « caché » et « regard » sur l'affichage de l'arête « aptF1F2 ».

Si la case « caché » n'est pas cochée
| alors l'arête « aptF1F2 » doit être invisible
| Sinon si la case « regard » n'est pas cochée
| | Alors l'arête « aptF1F2 » doit être visible ou invisible selon que les valeurs des
| | indicateurs « indFa1 » et indFa2 » sont simultanément « False » ou non
| | Sinon l'arête « aptF1F2 » doit être visible ou invisible selon que les valeurs des
| | indicateurs « indFa1 » et indFa2 » sont simultanément « True » ou non
| Fin si alors sinon
Fin si alors sinon

Dans la fenêtre Algèbre, sélectionnons les segments « aptF1F2 », « aptF1F3 », « aptF1F4 », « aptF1F5 », « aptF2F3 », « aptF2F5 », « aptF2F6 », « aptF3F4 », « aptF3F6 », « aptF4F5 », « aptF4F6 » et « aptF5F6 » (avec les touches « control » et « majuscules » pour faire des sélections multiples), puis, dans Propriétés, onglet Avancé, conditions pour afficher l'objet, écrivons la formule suivante :

Si[¬caché, false, Si[¬regard, (¬indFa1) ∧ (¬indFa2), indFa1 ∧ indFa2]]

Dans la fenêtre Algèbre, il suffira maintenant de sélectionner uniquement l'arête « aptF1F3 » puis, dans Propriétés, onglet Avancé, conditions pour afficher l'objet, nous trouverons la formule prévue pour l'arête précédente

(Si[¬caché, false, Si[¬regard, (¬indFa1) ∧ (¬indFa2), indFa1 ∧ indFa2]])

dans laquelle il suffira simplement de modifier 2 caractères :

Si[¬caché, false, Si[¬regard, (¬indFa1) ∧ (¬indFa3), indFa1 ∧ indFa3]].

On traitera les arêtes cachées restantes de la même manière et on leur donnera le style trait pointillés, épaisseur 3. Nous n'afficherons plus les étiquettes des 24 segments.

Paramétrons les arêtes vues.

Une arête est vue lorsqu'elle n'est pas cachée. Nous utiliserons donc, pour établir « aplF1F2 », les négations des conditions utilisées pour établir « aptF1F2 », c'est-à-dire

(¬indFa1) ∧ (¬indFa2) et indFa1 ∧ indFa2

Or ¬[(¬indFa1) ∧ (¬indFa2)] se simplifie en ¬(¬indFa1) ∨ ¬(¬indFa2) donc en indFa1 ∨ indFa2

et ¬[indFa1 ∧ indFa2] se simplifie en (¬indFa1) ∨ (¬indFa2).

Dans la fenêtre Algèbre, sélectionnons les segments « aplF1F2 », « aplF1F3 », « aplF1F4 », « aplF1F5 », « aplF2F3 », « aplF2F5 », « aplF2F6 », « aplF3F4 », « aplF3F6 », « aplF4F5 », « aplF4F6 » et « aplF5F6 » (avec les touches « control » et « majuscules » pour faire des sélections multiples), puis, dans Propriétés, onglet Avancé, conditions pour afficher l'objet, écrivons la formule suivante : Si[¬regard, indFa1 ∨ indFa2, (¬indFa1) ∨ (¬indFa2)]

Dans la fenêtre Algèbre, il suffira maintenant de sélectionner uniquement l'arête « aplF1F3 » puis, dans Propriétés, onglet Avancé, conditions pour afficher l'objet, nous trouverons la formule prévue pour l'arête précédente (Si[¬regard, indFa1 ∨ indFa2, (¬indFa1) ∨ (¬indFa2)]) dans laquelle il suffira simplement de modifier 2 caractères : Si[¬regard, indFa1 ∨ indFa3, (¬indFa1) ∨ (¬indFa3)].

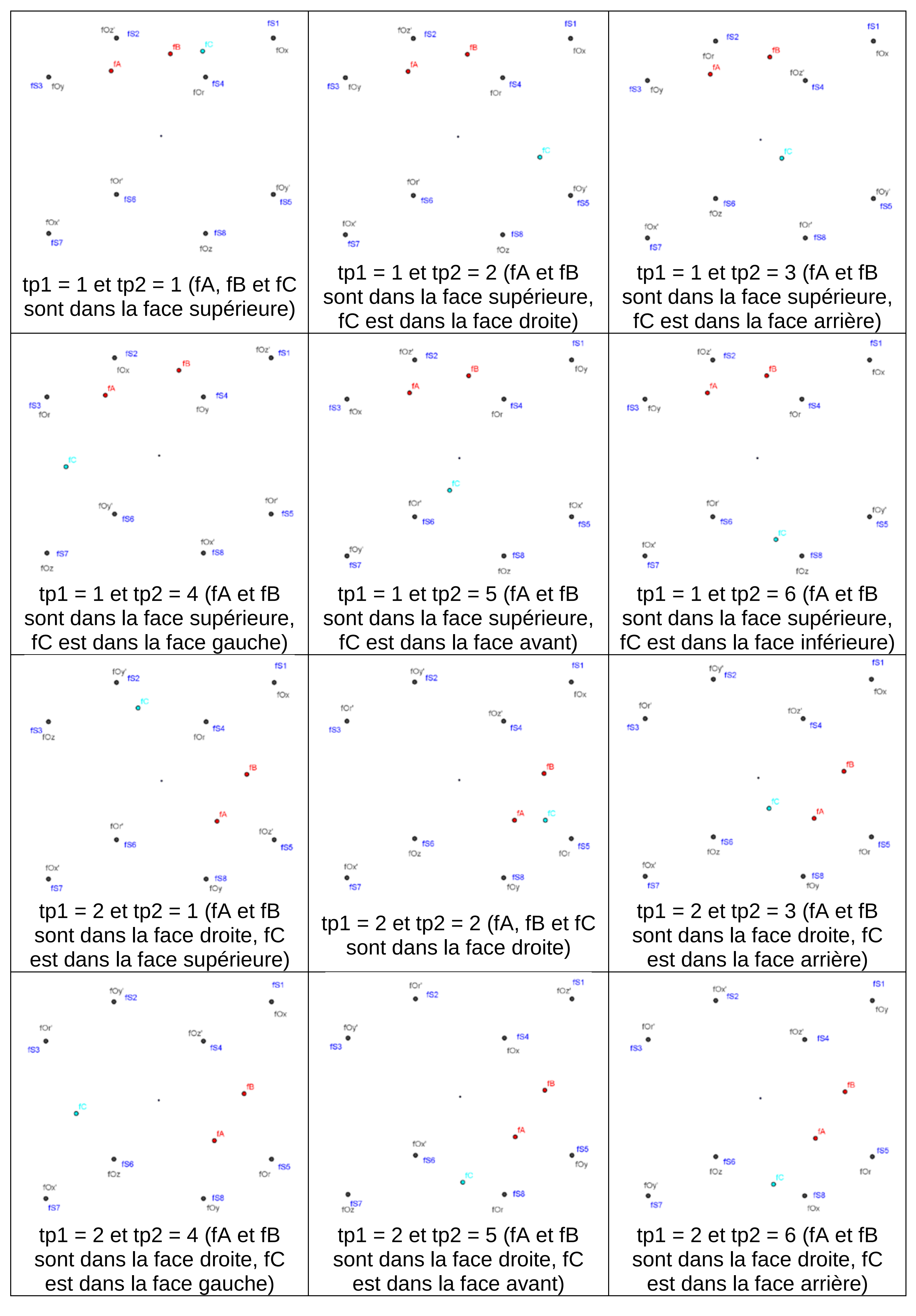
On traitera les arêtes vues restantes de la même manière et on leur donnera le style trait plein, épaisseur 4, en couleur noire.

Paramétrons les sommets.

Dans la fenêtre Algèbre, sélectionnons les points « mS1 », « mS2 », « mS3 », « mS4 », « mS5 », « mS6 », « mS7 » et « mS8 » (avec les touches « control » et « majuscules » pour faire des sélections multiples), puis, dans Propriétés, onglet Avancé, conditions pour afficher l'objet, écrivons la condition suivante : sommets.

Des points placés sur les sommets pour supporter les constructions !

Pour pouvoir assurer la suite des constructions, nous utiliserons des sommets supports « fOr », « fOx », « fOy », « fOz » et leurs symétriques par rapport à « Poignée », « fOr' », « fOx' », « fOy' » et « fOz' ». Ils se placeront sur les sommets du cube en fonction des valeurs prises par « tp1 » et « tp2 » en tenant compte du fait que « fOr « et « fOx » seront les sommets du cube communs à la face contenant « fA » et « fB » et à la face contenant « fC » lorsque ces faces ne sont ni opposées ni confondues, en tenant compte du fait que « fOy » sera le sommet qui est opposé à « fOx » dans la face contenant « fA » et « fB » et en tenant compte du fait que « fOz » sera le sommet qui est opposé à « fOx » dans la face contenant « fC ». Le tableau suivant montre leurs positions.



tp1 = 1 et tp2 = 1 (fA, fB et fC sont dans la face supérieure)

tp1 = 1 et tp2 = 2 (fA et fB sont dans la face supérieure, fC est dans la face droite)

tp1 = 1 et tp2 = 3 (fA et fB sont dans la face supérieure, fC est dans la face arrière)

tp1 = 1 et tp2 = 4 (fA et fB sont dans la face supérieure, fC est dans la face gauche)

tp1 = 1 et tp2 = 5 (fA et fB sont dans la face supérieure, fC est dans la face avant)

tp1 = 1 et tp2 = 6 (fA et fB sont dans la face supérieure, fC est dans la face inférieure)

tp1 = 2 et tp2 = 1 (fA et fB sont dans la face droite, fC est dans la face supérieure)

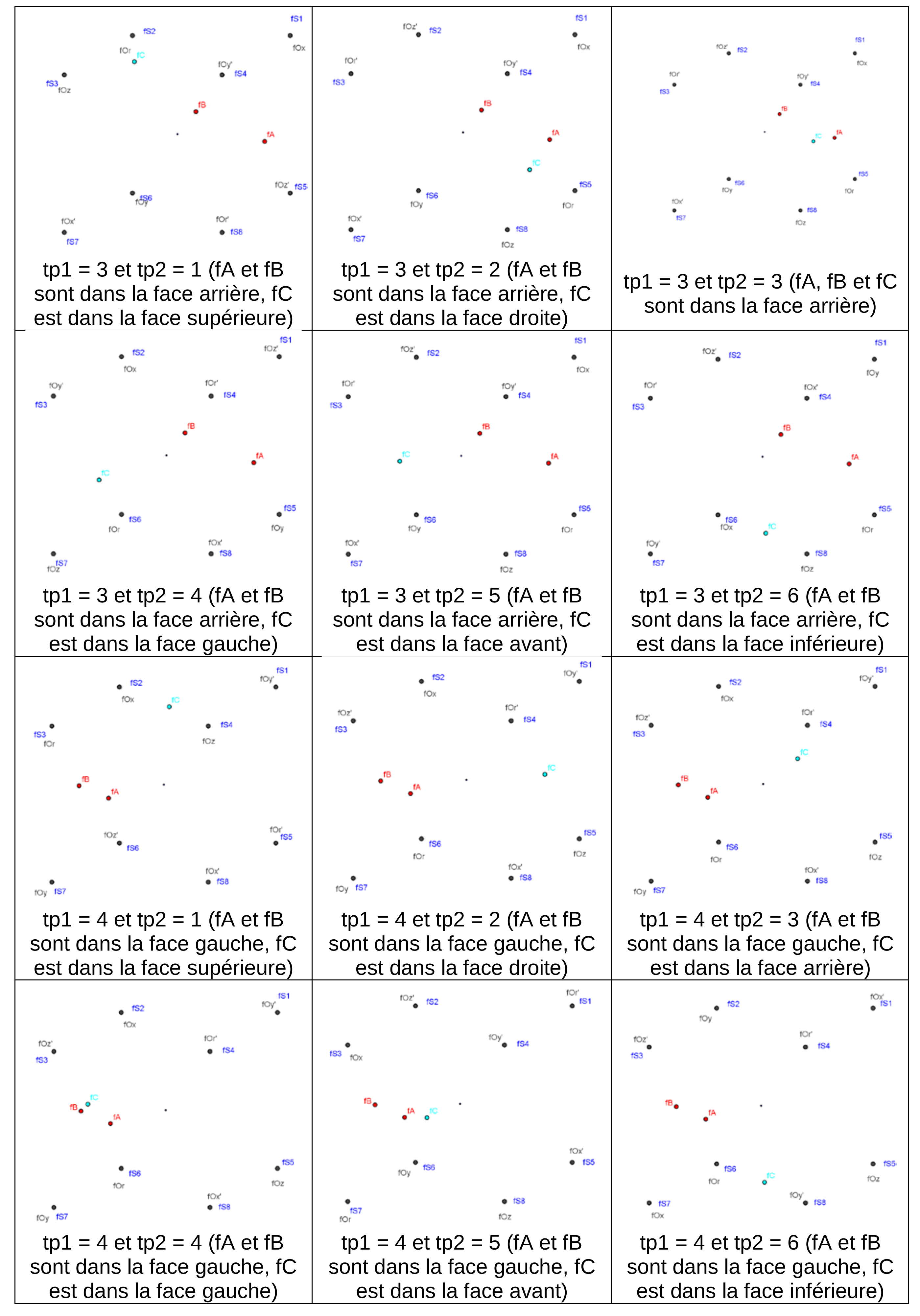
tp1 = 2 et tp2 = 2 (fA, fB et fC sont dans la face droite)

tp1 = 2 et tp2 = 3 (fA et fB sont dans la face droite, fC est dans la face arrière)

tp1 = 2 et tp2 = 4 (fA et fB sont dans la face droite, fC est dans la face gauche)

tp1 = 2 et tp2 = 5 (fA et fB sont dans la face droite, fC est dans la face avant)

tp1 = 2 et tp2 = 6 (fA et fB sont dans la face droite, fC est dans la face arrière)



tp1 = 3 et tp2 = 1 (fA et fB sont dans la face arrière, fC est dans la face supérieure)

tp1 = 3 et tp2 = 2 (fA et fB sont dans la face arrière, fC est dans la face droite)

tp1 = 3 et tp2 = 3 (fA, fB et fC sont dans la face arrière)

tp1 = 3 et tp2 = 4 (fA et fB sont dans la face arrière, fC est dans la face gauche)

tp1 = 3 et tp2 = 5 (fA et fB sont dans la face arrière, fC est dans la face avant)

tp1 = 3 et tp2 = 6 (fA et fB sont dans la face arrière, fC est dans la face inférieure)

tp1 = 4 et tp2 = 1 (fA et fB sont dans la face gauche, fC est dans la face supérieure)

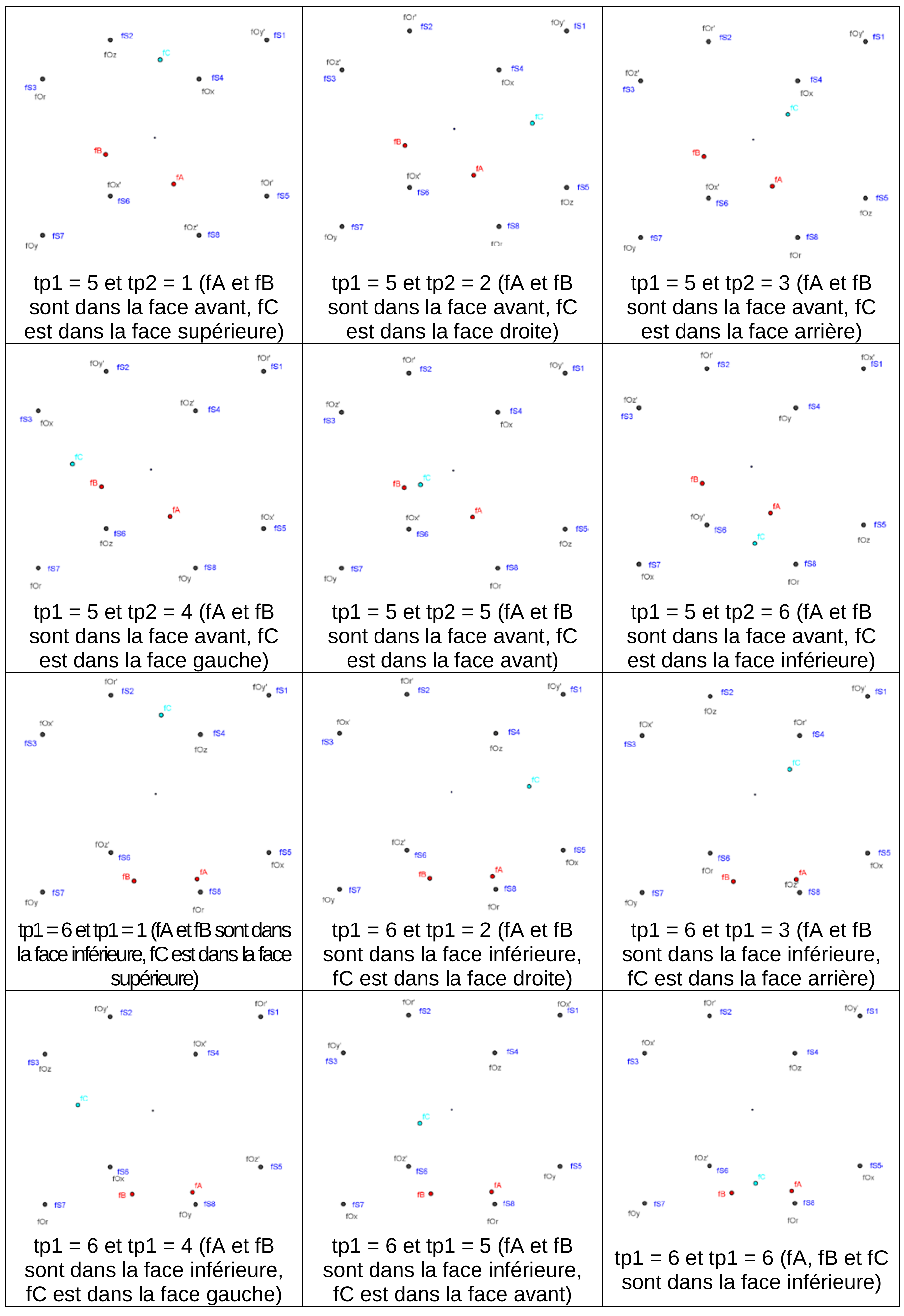
tp1 = 4 et tp2 = 2 (fA et fB sont dans la face gauche, fC est dans la face droite)

tp1 = 4 et tp2 = 3 (fA et fB sont dans la face gauche, fC est dans la face arrière)

tp1 = 4 et tp2 = 4 (fA et fB sont dans la face gauche, fC est dans la face gauche)

tp1 = 4 et tp2 = 5 (fA et fB sont dans la face gauche, fC est dans la face avant)

tp1 = 4 et tp2 = 6 (fA et fB sont dans la face gauche, fC est dans la face inférieure)



tp1 = 5 et tp2 = 1 (fA et fB sont dans la face avant, fC est dans la face supérieure)

tp1 = 5 et tp2 = 2 (fA et fB sont dans la face avant, fC est dans la face droite)

tp1 = 5 et tp2 = 3 (fA et fB sont dans la face avant, fC est dans la face arrière)

tp1 = 5 et tp2 = 4 (fA et fB sont dans la face avant, fC est dans la face gauche)

tp1 = 5 et tp2 = 5 (fA et fB sont dans la face avant, fC est dans la face avant)

tp1 = 5 et tp2 = 6 (fA et fB sont dans la face avant, fC est dans la face inférieure)

tp1 = 6 et tp1 = 1 (fA et fB sont dans la face inférieure, fC est dans la face supérieure)

tp1 = 6 et tp1 = 2 (fA et fB sont dans la face inférieure, fC est dans la face droite)

tp1 = 6 et tp1 = 3 (fA et fB sont dans la face inférieure, fC est dans la face arrière)

tp1 = 6 et tp1 = 4 (fA et fB sont dans la face inférieure, fC est dans la face gauche)

tp1 = 6 et tp1 = 5 (fA et fB sont dans la face inférieure, fC est dans la face avant)

tp1 = 6 et tp1 = 6 (fA, fB et fC sont dans la face inférieure)

Voici un schéma expliquant la formule permettant d'obtenir « fOr » au bon endroit.

```
Si tp1 = 1
| Alors  Si tp2 = 3
|        | alors fOr est sur fS2
|        | sinon  Si tp2 = 4
|        |        | alors fOr est sur fS3
|        |        | sinon  fOr est sur fS4
|        |        Fin si alors sinon
|        Fin si alors sinon
| Sinon  Si tp1 = 2
|        | Alors  Si tp2 = 1
|        |        | Alors fOr est sur fS4
|        |        | Sinon  Si tp2 = 5
|        |        |        | Alors fOr est sur fS8
|        |        |        | Sinon fOr est sur fS5
|        |        |        Fin si alors sinon
|        |        Fin si alors sinon
|        | Sinon  Si tp1 = 3
|        |        | Alors  Si tp2 = 1
|        |        |        | Alors fOr est sur fS2
|        |        |        | Sinon  Si tp2 = 4
|        |        |        |        | Alors fOr est sur fS6
|        |        |        |        | Sinon fOr est sur fS5
|        |        |        |        Fin si alors sinon
|        |        |        Fin si alors sinon
|        |        | Sinon  Si tp1 = 4
|        |        |        | Alors  Si tp2 = 1
|        |        |        |        | Alors fOr est sur fS3
|        |        |        |        | Sinon  Si tp2 = 5
|        |        |        |        |        | Alors fOr est sur fS7
|        |        |        |        |        | Sinon fOr est sur fS6
|        |        |        |        |        Fin si alors sinon
|        |        |        |        Fin si alors sinon
|        |        |        | Sinon  Si tp1 = 5
|        |        |        |        | Alors  Si tp2 = 1
|        |        |        |        |        | Alors fOr est sur fS3
|        |        |        |        |        | Sinon  Si tp2 = 4
|        |        |        |        |        |        | Alors fOr est sur fS7
|        |        |        |        |        |        | Sinon fOr est sur fS8
|        |        |        |        |        |        Fin si alors sinon
|        |        |        |        |        Fin si alors sinon
|        |        |        |        | Sinon  Si tp2 = 3
|        |        |        |        |        | Alors fOr est sur fS6
|        |        |        |        |        | Sinon  Si tp2 = 4
|        |        |        |        |        |        | Alors fOr est sur fS7
|        |        |        |        |        |        | Sinon fOr est sur fS8
|        |        |        |        |        |        Fin si alors sinon
|        |        |        |        |        Fin si alors sinon
|        |        |        |        Fin si alors sinon
|        |        |        Fin si alors sinon
|        |        Fin si alors sinon
|        Fin si alors sinon
Fin si alors sinon
```

40) Saisie : fOr = Si[tp1 ≟ 1, Si[tp2 ≟ 3, fS2, Si[tp2 ≟ 4, fS3, fS4]], Si[tp1 ≟ 2, Si[tp2 ≟ 1, fS4, Si[tp2 ≟ 5, fS8, fS5]], Si[tp1 ≟ 3, Si[tp2 ≟ 1, fS2, Si[tp2 ≟ 4, fS6, fS5]], Si[tp1 ≟ 4, Si[tp2 ≟ 1, fS3, Si[tp2 ≟ 5, fS7, fS6]], Si[tp1 ≟ 5, Si[tp2 ≟ 1, fS3, Si[tp2 ≟ 4, fS7, fS8]], Si[tp2 ≟ 3, fS6, Si[tp2 ≟ 4, fS7, fS8]]]]]]

41) Symétrie centrale : fOr' = Symétrie[fOr, Poignée]

De manière analogue, on a

42) Saisie : fOx = Si[tp1 ≟ 1, Si[tp2 ≟ 4, fS2, Si[tp2 ≟ 5, fS3, fS1]], Si[tp1 ≟ 2, Si[tp2 ≟ 5, fS4, Si[tp2 ≟ 6, fS8, fS1]], Si[tp1 ≟ 3, Si[tp2 ≟ 4, fS2, Si[tp2 ≟ 6, fS6, fS1]], Si[tp1 ≟ 4, Si[tp2 ≟ 5, fS3, Si[tp2 ≟ 6, fS7, fS2]], Si[tp1 ≟ 5, Si[tp2 ≟ 4, fS3, Si[tp2 ≟ 6, fS7, fS4]], Si[tp2 ≟ 4, fS6, Si[tp2 ≟ 5, fS7, fS5]]]]]]

43) Symétrie centrale : fOx' = Symétrie[fOx, Poignée]

44) Saisie : fOy = Si[tp1 ≟ 1, Si[tp2 ≟ 4, fS4, Si[tp2 ≟ 5, fS1, fS3]], Si[tp1 ≟ 2, Si[tp2 ≟ 5, fS5, Si[tp2 ≟ 6, fS1, fS8]], Si[tp1 ≟ 3, Si[tp2 ≟ 4, fS5, Si[tp2 ≟ 6, fS1, fS6]], Si[tp1 ≟ 4, Si[tp2 ≟ 5, fS6, Si[tp2 ≟ 6, fS2, fS7]], Si[tp1 ≟ 5, Si[tp2 ≟ 4, fS8, Si[tp2 ≟ 6, fS4, fS7]], Si[tp2 ≟ 4, fS8, Si[tp2 ≟ 5, fS5, fS7]]]]]]

45) Symétrie centrale : fOy' = Symétrie[fOy, Poignée]

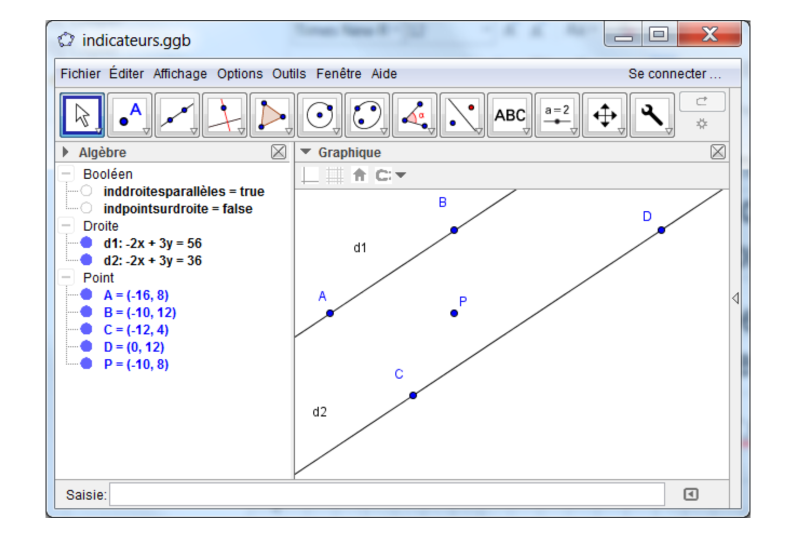
46) Saisie : fOz = Si[tp1 ≟ 1, Si[tp2 ≟ 3, fS6, Si[tp2 ≟ 4, fS7, fS8]], Si[tp1 ≟ 2, Si[tp2 ≟ 1, fS3, Si[tp2 ≟ 5, fS7, fS6]], Si[tp1 ≟ 3, Si[tp2 ≟ 1, fS3, Si[tp2 ≟ 4, fS7, fS8]], Si[tp1 ≟ 4, Si[tp2 ≟ 1, fS4, Si[tp2 ≟ 5, fS8, fS5]], Si[tp1 ≟ 5, Si[tp2 ≟ 1, fS2, Si[tp2 ≟ 4, fS6, fS5]], Si[tp2 ≟ 3, fS2, Si[tp2 ≟ 4, fS3, fS4]]]]]]

47) Symétrie centrale : fOz' = Symétrie[fOz, Poignée]

Nous masquerons tous ces points.

Petite parenthèse

Pour la suite de la construction de la section, nous rencontrerons le besoin de savoir si deux droites sont parallèles ou non, si un point appartient à une droite ou non. Nous allons donc faire une petite pause dans la recherche de notre section pour élaborer les outils permettant de répondre à ces besoins.



Créons le fichier « indicateur.ggb ».

1) Point : A

2) Point : B

3) Point : C

4) Point : D

5) Point : P

6) Droite : d1 = Droite[A, B]

7) Droite : d2 = Droite[C, D]

d1 et d2 seront parallèles lorsque l'image de d2 par la translation appliquant C sur A sera confondue avec d1. On créera donc l'indicateur suivant :

8) Booléen: inddroitesparallèles = Translation[d2, Vecteur[C, A]] ≟ d1

P et d1 seront parallèles lorsque la parallèle à d1 comprenant P sera confondue avec d1. On créera donc l'indicateur suivant :

9) Booléen: indpointsurdroite = Droite[P, d1] ≟ d1

Une macro pour déceler le parallélisme de deux droites

Pour créer la macro « droitesparallèles.ggt »	
4 Objets initiaux :	**1 Objet final :**
A, d1, C et d2	inddroitesparallèles
Aide :	
Désignez un point de d1, d1, un point de d2 puis d2. La macro donne un booléen dont la valeur est « true » ou « false » selon que d1 et d2 sont parallèles ou ne le sont pas.	

Une macro pour déceler l'appartenance d'un point à une droite

Pour créer la macro « pointsurdroite.ggt »	
2 Objets initiaux :	**1 Objet final :**
P, d1	indpointsurdroite
Aide :	
Désignez le point P puis la droite d. La macro donne un booléen dont la valeur est « true » ou « false » selon que P est ou non sur d.	

Une macro pour déterminer la trace d'une droite dans une face : « tracedansface.ggb »

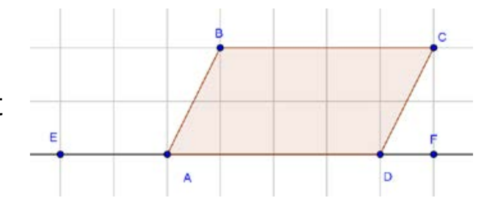
Vous allez probablement trouver inutile une telle macro puisque Geogebra permet de construire les points d'intersection d'un quadrilatère et d'une droite et qu'il suffirait dès lors de construire le segment déterminé par ces points.

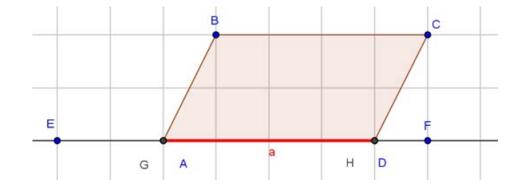
Et pourtant !

Lorsque Geogebra construit l'intersection d'un quadrilatère et d'une droite, il considère quatre points même s'il n'en montre parfois que deux. C'est logique puisque l'intersection d'un quadrilatère convexe et d'une droite donne, au plus, un seul segment alors que l'intersection d'un quadrilatère non convexe et d'une droite peut en donner deux.

Dans certaines configurations particulières, les deux points inattendus peuvent réapparaître.

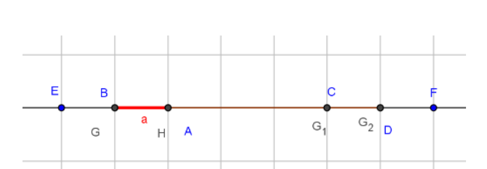
Pour s'en rendre compte, construisez le quadrilatère ABCD et la droite EF en vous servant de la grille.





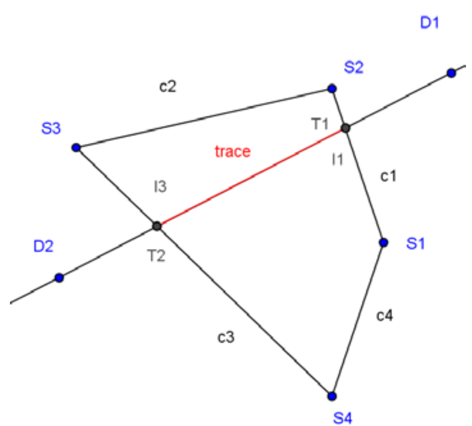
Construisez ensuite l'intersection du quadrilatère et de la droite. Geogebra donne les points G et H que vous relierez par un segment rouge.

Si vous déplacez le segment BC de manière à placer le point A comme le montre la figure suivante, vous serez surpris deux fois ! Non seulement Geogebra fait apparaître deux points G1 et G2, ce qui n'est pas très grave, mais, et cela est embêtant, il décide que les extrémités du segment GH qui coïncidaient avec A et D coïncideront désormais avec A et B.



Créons donc une macro pour pallier ces inconvénients.

Créons le fichier **« <u>tracedansface.ggb</u> »**



Nous créerons les points « S1 », « S2 », « S3 » et « S4 » et les segments déterminés par ces points, « c1 », « c2 », « c3 » et « c4 » et créons une droite « d » déterminée par les points « D1 » et « D2 ».

Nous construirons les points « I1 », « I2 », « I3 » et « I4 », communs à « d » et à, respectivement, « c1 », « « c2 », « c3 » et « c4 ».

Nous placerons le point « T1 » sur « I1 » si ce dernier est défini ou sur « I2 » si ce dernier est défini ou sur « I3 » si ce dernier est défini ou sur « I4 » si ce dernier est défini.

Nous placerons le point « T2 » sur « I4 » si ce dernier est défini et est différent de « I1 » ou sur « I3 » si ce dernier est défini ou sur « I2 » si ce dernier est défini ou sur « I1 » si ce dernier est défini.

Pour éviter les imprécisions dans le cas particulier où la droite est pratiquement parallèle à un côté de la face, nous construirons ensuite le segment « trace » de la manière suivante.
Si « d » coïncide avec S1S2, « trace » sera « c1 », si « d » coïncide avec S2S3, « trace » sera « c2 », si « d » coïncide avec S3S4, « trace » sera « c3 », si « d » coïncide avec S1S4, « trace » sera « c4 » et si aucun des cas précédents ne se présente, « trace » aura pour extrémités, s'ils sont définis, les points « T1 » et « T2 » à condition qu'ils soient distants l'un de l'autre de plus de 0.0000001 pour contourner les imprécisions de calcul qui provoqueraient l'existence d'un segment de longueur quasi nulle.

Si nous avions placé le point « T2 » sur « I4 » sans prendre la précaution que ce dernier soit défini <u>et soit différent de « I1 »</u>, le segment trace aurait été réduit à deux points confondus dans le cas où « d » aurait été la diagonale S1S3 du quadrilatère convexe S1S2S3S4.

1) Point : S1
2) Point : S2
3) Point : S3
4) Point : S4
5) Segment : c1 = Segment[S1, S2]
6) Segment : c2 = Segment[S2, S3]
7) Segment : c3 = Segment[S3, S4]
8) Segment : c4 = Segment[S4, S1]
9) Point : D1
10) Point : D2
11) Droite : d = Droite[D1, D2]
12) Intersection : I1 = Intersection[c1, d]
13) Intersection : I2 = Intersection[c2, d]
14) Intersection : I3 = Intersection[c3, d]
15) Intersection : I4 = Intersection[c4, d]
16) Saisie : T1 = Si[EstDéfini[I1], I1, Si[EstDéfini[I2], I2, Si[EstDéfini[I3], I3, I4]]]
17) Saisie : T2 = Si[EstDéfini[I4] ∧ (I4 ≠ I1), I4, Si[EstDéfini[I3], I3, Si[EstDéfini[I2], I2, I1]]]
18) Segment : trace = Si[Droite[S1, S2] ≟ d, c1, Si[Droite[S2, S3] ≟ d, c2, Si[Droite[S3, S4] ≟ d, c3, Si[Droite[S1, S4] ≟ d, c4, Si[Distance[T1, T2] > 0.0000001, Segment[T1, T2]]]]]]

<u>Pour créer la macro « tracedansface.ggt »</u>

<u>5 Objets initiaux :</u>

d, S1, S2, S3, S4

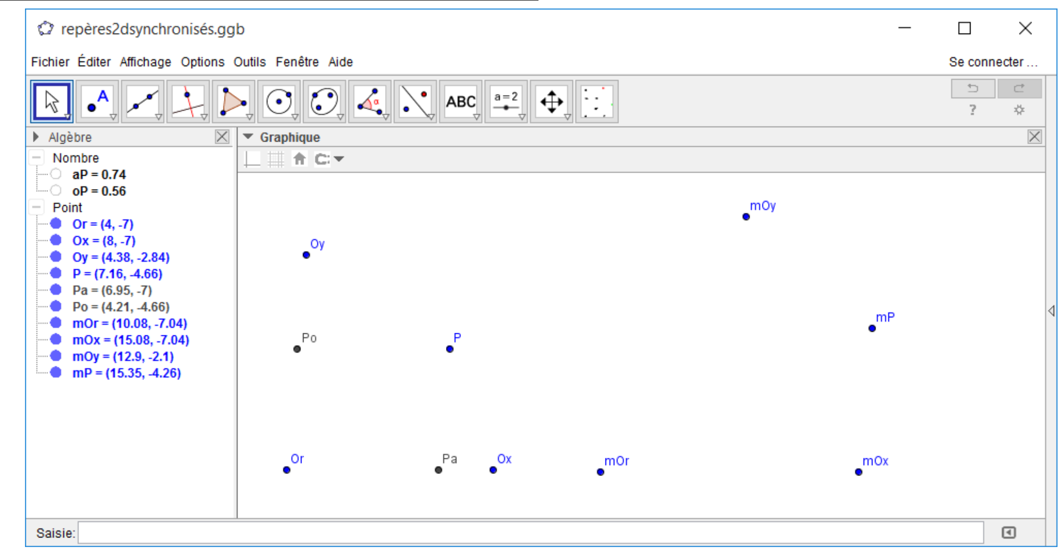
<u>1 Objet final :</u>

trace

<u>Aide :</u>

Désignez la droite puis les sommets de la face du cube. La macro donnera le segment montrant la trace de la droite dans la face du cube.

Report des constructions du cube fixe au cube mobile



Créons une macro qui nous donnera, dans un repère (mOr, mOx, mOy) donné, un point « mP » placé comme l'est le point « P » donné dans un repère (Or, Ox, Oy) également donné.

Pour ce faire, nous construirons « Pa », la projection de « P » sur la droite OrOx parallèlement à la direction OrOy et nous construirons « Po », la projection de « P » sur la droite OrOy parallèlement à la direction OrOx.

Nous déterminerons ensuite « aP », le rapport de l'homothétie de centre « Or» appliquant « Ox » sur « Pa » ainsi que « oP », le rapport de l'homothétie de centre « fOr» appliquant « Oy » sur « Po ».

Pour ce faire, rappelons que la norme d'un vecteur est la racine carrée du produit scalaire de ce vecteur par lui-même et rappelons que le produit scalaire des vecteurs $\vec{u}$ et $\vec{v}$ est égal au produit de leurs normes par le cosinus de l'angle qu'ils déterminent. Si $\vec{u}$ et $\vec{v}$ sont de même sens, l'angle qu'ils déterminent sera 0° et le signe de leur produit scalaire sera donc positif puisque cos 0° = 1 mais si $\vec{u}$ et $\vec{v}$ sont de sens contraires, l'angle qu'ils déterminent sera 180° et le signe de leur produit scalaire sera donc négatif puisque cos 180° = -1.

Soit C, O et I trois points alignés distincts deux à deux, soit $\vec{u}$ le vecteur d'origine C et d'extrémité I et soit $\vec{v}$ le vecteur d'origine C et d'extrémité O. Le rapport de l'homothétie de centre C appliquant O sur I est donc le rapport des normes des vecteurs $\vec{u}$ et $\vec{v}$ affecté du signe de leur produit scalaire ce qui donne, en langage Geogebra, sgn(u v) sqrt(u u) / sqrt(v v)

Le point « mP » sera alors l'image de « mOr » par la translation dont le vecteur sera la somme du vecteur égal au produit de « aP » par le vecteur appliquant « mOr » sur « mOx » et du vecteur égal au produit de « oP » par le vecteur appliquant « mOr » sur « mOy ».

Créons le fichier **« <u>repères2dsynchronisés.ggb</u> »**

1) Point : Or
2) Point : Ox
3) Point : Oy
4) Point : P
5) Point : mOr
6) Point : mOx
7) Point : mOy

8) Saisie : Pa = Intersection[Droite[P, Droite[Or, Oy]], Droite[Or, Ox]]
9) Saisie : Po = Intersection[Droite[P, Droite[Or, Ox]], Droite[Or, Oy]]
10) Saisie : aP = sgn(Vecteur[Or, Pa] Vecteur[Or, Ox]) sqrt(Vecteur[Or, Pa] Vecteur[Or, Pa]) / sqrt(Vecteur[Or, Ox] Vecteur[Or, Ox])
11) Saisie : oP = sgn(Vecteur[Or, Po] Vecteur[Or, Oy]) sqrt(Vecteur[Or, Po] Vecteur[Or, Po]) / sqrt(Vecteur[Or, Oy] Vecteur[Or, Oy])
12) Saisie : mP = Translation[mOr, Vecteur[aP Vecteur[mOr, mOx] + oP Vecteur[mOr, mOy]]]

Pour créer la macro « repères2dsynchronisés.ggt »	
7 Objets initiaux :	**1 Objet final :**
P, Or, Ox, Oy, mOr, mOx et mOy	mP
Aide :	
Cliquez sur P, Or, Ox, Oy, mOr, mOx et mOy. La macro donnera mP.	

Revenons à la construction de notre section.

Remarquons que, fA, fB et fC doivent être distincts deux à deux et non alignés.

Créons « ind3pdis », un indicateur

- prenant la valeur true lorsque les trois points fA, fB et fC, placés dans une même face ou placés sur l'intersection de deux faces, sont distincts deux à deux ou lorsque les deux points fA et fB, placés dans une autre face que celle contenant fC, sont distincts
- prenant la valeur « false » dans les autres cas.

Si fA, fB et fC sont situés dans une même face ou si fC est situé sur la droite fOrfOx
| Alors la valeur de ind3pdis est la valeur de la conjonction des trois conditions
| « fA est distinct de fB », « fA distinct de fC » et « fB distinct de fC »
| Sinon la valeur de ind3pdis est la valeur de la condition « fA est distinct de fB »
Fin si alors sinon

Remarquons donc que cet indicateur indiquera que les points fA et fC, chacun distinct de fC et placés dans deux faces différentes en dehors de l'intersection de ces deux faces, seront distincts même s'ils coïncident à l'écran. Il en va de même pour les points fB et fC.

Ouvrons l'outil « pointsurdroite.ggt ».

48) Saisie : ind3pdis = Si[tp1 ≟ tp2 ∨ pointsurdroite[fC, Droite[fOr, fOx]], (fA ≠ fB) ∧ (fA ≠ fC) ∧ (fB ≠ fC), fA ≠ fB]

Créons un texte d'avertissement de points non distincts.

49) Texte : textepointsdistincts = ''La section ne peut être déterminée car vos trois points ne sont pas distincts deux à deux.'' (Propriétés, onglet Avancé, Conditions pour afficher l'objet : ¬ind3pdis)

Créons « ind3pali », un indicateur

- prenant la valeur true lorsque les trois points fA, fB et fC, placés dans une même face ou placés sur l'intersection de deux faces sont situés de telle manière qu'une au moins des conditions suivantes suivante soit vérifiée : « la distance de fA à fB égale la somme des distances de fA à fC et de fC à fB », « la distance de fA à fC égale la somme des distances de fA à fB et de fB à fC » et « la distance de fB à fC égale la somme des distances de fB à fA et de fA à fC »
- prenant la valeur « false » dans les autres cas.

Si fA, fB et fC sont distincts et que,
| soit fA, fB et fC sont situés dans une même face, soit fC est situé sur la droite fOrfOx
| Alors la valeur de ind3pdis est la valeur de la disjonction des trois conditions
| « la distance de fA à fB égale la somme des distances de fA à fC et de fC à fB »
| « la distance de fA à fC égale la somme des distances de fA à fB et de fB à fC »
| « la distance de fB à fC égale la somme des distances de fB à fA et de fA à fC »
Fin si alors sinon

50) Saisie : ind3pali = ind3pdis ∧ Si[tp1 ≟ tp2 ∨ pointsurdroite[fC, Droite[fOr, fOx]], Distance[fA, fB] ≟ Distance[fA, fC] + Distance[fC, fB] ∨ Distance[fA, fC] ≟ Distance[fA, fB] + Distance[fB, fC] ∨ Distance[fB, fC] ≟ Distance[fB, fA] + Distance[fA, fC]]

51) Texte : textepointsalignés = ‘’La section ne peut être déterminée car vos trois points sont alignés.’’ (Propriétés, onglet Avancé, Conditions pour afficher l’objet : ind3pali)

<u>Reportons les éléments utilisés pour les constructions du cube fixe dans le cube mobile</u>

Pour établir la formule donnant « mOr », nous reprendrons la formule donnant « fOr »

Si[tp1 ≟ 1, Si[tp2 ≟ 3, fS2, Si[tp2 ≟ 4, fS3, fS4]], Si[tp1 ≟ 2, Si[tp2 ≟ 1, fS4, Si[tp2 ≟ 5, fS8, fS5]], Si[tp1 ≟ 3, Si[tp2 ≟ 1, fS2, Si[tp2 ≟ 4, fS6, fS5]], Si[tp1 ≟ 4, Si[tp2 ≟ 1, fS3, Si[tp2 ≟ 5, fS7, fS6]], Si[tp1 ≟ 5, Si[tp2 ≟ 1, fS3, Si[tp2 ≟ 4, fS7, fS8]], Si[tp2 ≟ 3, fS6, Si[tp2 ≟ 4, fS7, fS8]]]]]]

dans laquelle nous remplacerons « fS2 » par « mS2 », « fS3 » par « mS2 », et cetera.

52) Saisie : mOr = Si[tp1 ≟ 1, Si[tp2 ≟ 3, mS2, Si[tp2 ≟ 4, mS3, mS4]], Si[tp1 ≟ 2, Si[tp2 ≟ 1, mS4, Si[tp2 ≟ 5, mS8, mS5]], Si[tp1 ≟ 3, Si[tp2 ≟ 1, mS2, Si[tp2 ≟ 4, mS6, mS5]], Si[tp1 ≟ 4, Si[tp2 ≟ 1, mS3, Si[tp2 ≟ 5, mS7, mS6]], Si[tp1 ≟ 5, Si[tp2 ≟ 1, mS3, Si[tp2 ≟ 4, mS7, mS8]], Si[tp2 ≟ 3, mS6, Si[tp2 ≟ 4, mS7, mS8]]]]]]

De manière analogue, on obtiendra

53) Saisie : mOx = Si[tp1 ≟ 1, Si[tp2 ≟ 4, mS2, Si[tp2 ≟ 5, mS3, mS1]], Si[tp1 ≟ 2, Si[tp2 ≟ 5, mS4, Si[tp2 ≟ 6, mS8, mS1]], Si[tp1 ≟ 3, Si[tp2 ≟ 4, mS2, Si[tp2 ≟ 6, mS6, mS1]], Si[tp1 ≟ 4, Si[tp2 ≟ 5, mS3, Si[tp2 ≟ 6, mS7, mS2]], Si[tp1 ≟ 5, Si[tp2 ≟ 4, mS3, Si[tp2 ≟ 6, mS7, mS4]], Si[tp2 ≟ 4, mS6, Si[tp2 ≟ 5, mS7, mS5]]]]]]

54) Saisie : mOy = Si[tp1 ≟ 1, Si[tp2 ≟ 4, mS4, Si[tp2 ≟ 5, mS1, mS3]], Si[tp1 ≟ 2, Si[tp2 ≟ 5, mS5, Si[tp2 ≟ 6, mS1, mS8]], Si[tp1 ≟ 3, Si[tp2 ≟ 4, mS5, Si[tp2 ≟ 6, mS1, mS6]], Si[tp1 ≟ 4, Si[tp2 ≟ 5, mS6, Si[tp2 ≟ 6, mS2, mS7]], Si[tp1 ≟ 5, Si[tp2 ≟ 4, mS8, Si[tp2 ≟ 6, mS4, mS7]], Si[tp2 ≟ 4, mS8, Si[tp2 ≟ 5, mS5, mS7]]]]]]

55) Saisie : mOz = Si[tp1 ≟ 1, Si[tp2 ≟ 3, mS6, Si[tp2 ≟ 4, mS7, mS8]], Si[tp1 ≟ 2, Si[tp2 ≟ 1, mS3, Si[tp2 ≟ 5, mS7, mS6]], Si[tp1 ≟ 3, Si[tp2 ≟ 1, mS3, Si[tp2 ≟ 4, mS7, mS8]], Si[tp1 ≟ 4, Si[tp2 ≟ 1, mS4, Si[tp2 ≟ 5, mS8, mS5]], Si[tp1 ≟ 5, Si[tp2 ≟ 1, mS2, Si[tp2 ≟ 4, mS6, mS5]], Si[tp2 ≟ 3, mS2, Si[tp2 ≟ 4, mS3, mS4]]]]]]

et donc 56) mOr’ = Symétrie[mOr, Poignée] 58) mOy’ = Symétrie[mOy, Poignée]
57) mOx’ = Symétrie[mOx, Poignée] 59) mOz’ = Symétrie[mOz, Poignée]

Remarquons que puisque « fA » et « fB » sont toujours situés dans le quadrilatère OrOxOyOz’, « mA » et « mB » seront toujours situés dans le quadrilatère mOrmOxmOymOz’.

Ouvrons l’outil « repères2dsynchronisés.ggt ».

60) Saisie : mA = repères2dsynchronisés[fA, fOr, fOx, fOy, mOr, mOx, mOy]

61) Saisie : mB = repères2dsynchronisés[fB, fOr, fOx, fOy, mOr, mOx, mOy]

Remarquons (page 9) que selon que « fA » et « fB » d’une part et « fC » d’autre part sont situés dans deux faces confondues, dans deux faces opposées ou dans deux faces sécantes, « fC » est situé dans le quadrilatère fOrfOxfOyfOz’, dans le quadrilatère fOr’fOx’fOy’fOz ou dans le quadrilatère fOrfOxfOy’fOz. Par conséquent, « mC » sera situé dans le quadrilatère mOrmOxmOymOz’, dans le quadrilatère mOr’mOx’mOy’mOz ou dans le quadrilatère mOrmOxmOy’mOz.

Nous devrons donc créer l’indicateur booléen « indfopo » dont la valeur sera « true » ou « false » selon que les faces contenant fA et fB pour l’une et fC pour l’autre seront ou non opposées.

62) Saisie : indfopo = tp1 ≟ 1 ∧ tp2 ≟ 6 ∨ tp1 ≟ 2 ∧ tp2 ≟ 4 ∨ tp1 ≟ 3 ∧ tp2 ≟ 5 ∨ tp1 ≟ 4 ∧ tp2 ≟ 2 ∨ tp1 ≟ 5 ∧ tp2 ≟ 3 ∨ tp1 ≟ 6 ∧ tp2 ≟ 1

63) Saisie : mC = Si[tp1 ≟ tp2, repères2dsynchronisés[fC, fOr, fOx, fOy, mOr, mOx, mOy], Si[indfopo, repères2dsynchronisés[fC, fOr', fOx', fOy', mOr', mOx', mOy'], repères2dsynchronisés[fC, fOr, fOx, fOz, mOr, mOx, mOz]]]

Etablissons un indicateur de visibilité pour le plan contenant mOr, mOx, mOy et mOz’

Il s’agit de la face supérieure lorsque le cube mobile est dans sa position de départ. Le plan contenant « mOr », « mOx », « mOz’ » et « mOy » se place toujours sur la face déterminée par « tp1 » et on peut donc établir le schéma suivant.

```
Si la case « regard » est cochée
|       Alors  Si tp1 = 1
|         |  alors indbssupérieure = contraire de indFa1
|         |  sinon  Si tp1 = 2
|         |         |  alors indbssupérieure = contraire de indFa2
|         |         |  sinon  Si tp1 = 3
|         |         |         | alors indbssupérieure = contraire de indFa3
|         |         |         | sinon  Si tp1 = 4
|         |         |         |        |  alors indbssupérieure = contraire de indFa4
|         |         |         |        |  sinon  Si tp1 = 5
|         |         |         |        |         |  alors indbssupérieure = contraire de indFa5
|         |         |         |        |         |  sinon indbssupérieure = contraire de indFa6
|         |         |         |        |         Fin alors sinon
|         |         |         |        Fin si alors sinon
|         |         |         Fin si alors sinon
|         |         Fin si alors sinon
|         Fin si alors sinon
| Sinon Si tp1 = 1
|         |  alors indbssupérieure = indFa1
|         |  sinon  Si tp1 = 2
|         |         |  alors indbssupérieure = indFa2
|         |         |  sinon  tp1 = 3
|         |         |         | alors indbssupérieure = indFa3
|         |         |         | sinon  Si tp1 = 4
|         |         |         |        |  alors indbssupérieure = indFa4
|         |         |         |        |  sinon  Si tp1 = 5
|         |         |         |        |         |  alors indbssupérieure = indFa5
|         |         |         |        |         |  sinon indbssupérieure = indFa6
|         |         |         |        |         Fin alors sinon
|         |         |         |        Fin si alors sinon
|         |         |         Fin si alors sinon
|         |         Fin si alors sinon
|         Fin si alors sinon
Fin si alors sinon
```

64) Saisie : indbssupérieure = Si[regard, Si[tp1 ≟ 1, ¬indFa1, Si[tp1 ≟ 2, ¬indFa2, Si[tp1 ≟ 3, ¬indFa3, Si[tp1 ≟ 4, ¬indFa4, Si[tp1 ≟ 5, ¬indFa5, ¬indFa6]]]]], Si[tp1 ≟ 1, indFa1, Si[tp1 ≟ 2, indFa2, Si[tp1 ≟ 3, indFa3, Si[tp1 ≟ 4, indFa4, Si[tp1 ≟ 5, indFa5, indFa6]]]]]]

Etablissons un indicateur de visibilité pour le plan contenant mOr’, mox’, mOy’ et mOz

On aura donc (pour la face inférieure lorsque le cube mobile est dans sa position de départ), les indicateurs de deux faces opposées étant chacun le contraire de l’autre,

65) Saisie : indbsinférieure = ¬indbssupérieure

<u>Etablissons un indicateur de visibilité pour le plan contenant mOr, mOx, mOy’ et mOz.</u>

Il s’agit de la face droite lorsque le cube mobile est dans sa position de départ. En observant le tableau de la page 9, on remarque que dans le cas où les faces contenant « mA » et « mB » pour l’une et « mC » pour l’autre seront opposées, le plan contenant « mOr », « mOx », « mOy’ » et « mOz » se place sur « Fa2 » lorsque la valeur de « tp2 » est égal à 1, 3, 5 ou 6 et se place sur « Fa3 » dans les autres cas. Dans le cas contraire, le plan contenant « mOr », « mOx », « mOy’ » et « mOz » se place sur la face désignée par la valeur de « tp2 ».

```
Si la case « regard » est cochée
| Alors  Si la face contenant fA et fB est opposée à la face contenant C
|        |  alors  Si tp1 = 1 ou tp1 = 3 ou tp1 = 5 ou tp1 = 6
|        |         |  alors indbsdroite = contraire de indFa2
|        |         |  sinon indbsdroite = contraire de indFa3
|        |         Fin si alors sinon
|        |  sinon  Si tp2 = 1
|        |         |  alors indbsdroite = contraire de indFa1
|        |         |  sinon  Si tp2 = 2
|        |         |         | alors indbsdroite = contraire de indFa2
|        |         |         | sinon  Si tp1 = 3
|        |         |         |        |  alors indbsdroite = contraire de indFa3
|        |         |         |        |  sinon  Si tp1 = 4
|        |         |         |        |         |  alors indbsdroite = contraire de indFa4
|        |         |         |        |         |  Sinon  Si tp1 = 5
|        |         |         |        |         |         | Alors indbsdroite = contraire de indFa5
|        |         |         |        |         |         | Sinon indbsdroite = contraire de indFa6
|        |         |         |        |         |         Fin alors sinon
|        |         |         |        |         Fin alors sinon
|        |         |         |        Fin si alors sinon
|        |         |         Fin si alors sinon
|        |         Fin si alors sinon
|        Fin si alors sinon
| Sinon  Si la face contenant fA et fB est opposée à la face contenant C
|        |  alors  Si tp1 = 1 ou tp1 = 3 ou tp1 = 5 ou tp1 = 6
|        |         |  alors indbsdroite = indFa2
|        |         |  sinon indbsdroite = indFa3
|        |         Fin si alors sinon
|        |  sinon  Si tp2 = 1
|        |         |  alors indbsdroite = indFa1
|        |         |  sinon  Si tp2 = 2
|        |         |         | alors indbsdroite = indFa2
|        |         |         | sinon  Si tp1 = 3
|        |         |         |        |  alors indbsdroite = indFa3
|        |         |         |        |  sinon  Si tp1 = 4
|        |         |         |        |         |  alors indbsdroite = indFa4
|        |         |         |        |         |  Sinon  Si tp1 = 5
|        |         |         |        |         |         | Alors indbsdroite = indFa5
|        |         |         |        |         |         | Sinon indbsdroite = indFa6
|        |         |         |        |         |         Fin alors sinon
|        |         |         |        |         Fin alors sinon
|        |         |         |        Fin si alors sinon
|        |         |         Fin si alors sinon
|        |         Fin si alors sinon
|        Fin si alors sinon
Fin si alors sinon
```

66) Saisie : indbsdroite = Si[regard, Si[indfopo, Si[tp1 ≟ 1 ∨ tp1 ≟ 3 ∨ tp1 ≟ 5 ∨ tp1 ≟ 6, ¬indFa2, ¬indFa3], Si[tp2 ≟ 1, ¬indFa1, Si[tp2 ≟ 2, ¬indFa2, Si[tp2 ≟ 3, ¬indFa3, Si[tp2 ≟ 4, ¬indFa4, Si[tp2 ≟ 5, ¬indFa5, ¬indFa6]]]]]], Si[indfopo, Si[tp1 ≟ 1 ∨ tp1 ≟ 3 ∨ tp1 ≟ 5 ∨ tp1 ≟ 6, indFa2, indFa3], Si[tp2 ≟ 1, indFa1, Si[tp2 ≟ 2, indFa2, Si[tp2 ≟ 3, indFa3, Si[tp2 ≟ 4, indFa4, Si[tp2 ≟ 5, indFa5, indFa6]]]]]]]

<u>Etablissons un indicateur de visibilité pour le plan contenant mOr’, mOx’, mOy et mOz’.</u>

On aura donc (pour la face gauche lorsque le cube mobile est dans sa position de départ),

67) Saisie : indbsgauche = ¬indbsdroite

<u>Etablissons un indicateur de visibilité pour le plan contenant mOr, mOy, mOx’ et mOz.</u>

Il s’agit de la face avant lorsque le cube mobile est dans sa position de départ.

En observant le tableau de la page 9, on peut établir le schéma suivant.

```
Si la case « regard » est cochée
|  Alors  Si tp1 = 1 et tp2 = 5 ou tp1 = 3 et tp2 = 6 ou tp1 = 5 et tp2 = 6 ou tp1 = 6 et tp2 = 5
|         |  alors indbsavant est indFa2
|         |  sinon  Si tp1 = 2 et tp2 = 6 ou tp1 = 4 et tp2 = 6
|         |         |  alors indbsavant est indFa3
|         |         |  sinon  Si tp1 = 1 et tp2 = 3 ou tp1 = 3 et tp2 = 1
|         |         |         |  ou tp1 = 5 et tp2 = 1 ou tp1 = 6 et tp2 = 3
|         |         |         |  alors indbsavant est indFa4
|         |         |         |  sinon  Si tp1 = 1 et tp2 = 2 ou tp1 = 1 et tp2 = 4
|         |         |         |         |  ou tp1 = 1 et tp2 = 6 ou tp1 = 2 et tp2 = 1
|         |         |         |         |  ou tp1 = 4 et tp2 = 1 ou tp1 = 6 et tp2 = 1
|         |         |         |         |  ou tp1 = 6 et tp2 = 2 ou tp1 = 6 et tp2 = 4
|         |         |         |         |     alors indbsavant est indFa5
|         |         |         |         |     sinon indbsavant est indFa6
|         |         |         |         Fin si alors sinon
|         |         |         Fin si alors sinon
|         |         Fin si alors sinon
|         Fin si alors sinon
|  Sinon  Si tp1 = 1 et tp2 = 5 ou tp1 = 3 et tp2 = 6 ou tp1 = 5 et tp2 = 6 ou tp1 = 6 et tp2 = 5
|         |  alors indbsavant est indFa2
|         |  sinon  Si tp1 = 2 et tp2 = 6 ou tp1 = 4 et tp2 = 6
|         |         |  alors indbsavant est indFa3
|         |         |  sinon  Si tp1 = 1 et tp2 = 3 ou tp1 = 3 et tp2 = 1
|         |         |         |  ou tp1 = 5 et tp2 = 1 ou tp1 = 6 et tp2 = 3
|         |         |         |  alors indbsavant est indFa4
|         |         |         |  sinon  Si tp1 = 1 et tp2 = 2 ou tp1 = 1 et tp2 = 4
|         |         |         |         |  ou tp1 = 1 et tp2 = 6 ou tp1 = 2 et tp2 = 1
|         |         |         |         |  ou tp1 = 4 et tp2 = 1 ou tp1 = 6 et tp2 = 1
|         |         |         |         |  ou tp1 = 6 et tp2 = 2 ou tp1 = 6 et tp2 = 4
|         |         |         |         |     alors indbsavant est indFa5
|         |         |         |         |     sinon indbsavant est indFa6
|         |         |         |         Fin si alors sinon
|         |         |         Fin si alors sinon
|         |         Fin si alors sinon
|         Fin si alors sinon
Fin si alors
```

68) Saisie : indbsavant = Si[regard, Si[tp1 ≟ 1 ∧ tp2 ≟ 5 ∨ tp1 ≟ 3 ∧ tp2 ≟ 6 ∨ tp1 ≟ 5 ∧ tp2 ≟ 6 ∨ tp1 ≟ 6 ∧ tp2 ≟ 5, ¬indFa2, Si[tp1 ≟ 2 ∧ tp2 ≟ 6 ∨ tp1 ≟ 4 ∧ tp2 ≟ 6, ¬indFa3, Si[tp1 ≟ 1 ∧ tp2 ≟ 3 ∨ tp1 ≟ 3 ∧ tp2 ≟ 1 ∨ tp1 ≟ 5 ∧ tp2 ≟ 1 ∨ tp1 ≟ 6 ∧ tp2 ≟ 3, ¬indFa4, Si[tp1 ≟ 1 ∧

tp2 ≟ 2 ∨ tp1 ≟ 1 ∧ tp2 ≟ 4 ∨ tp1 ≟ 1 ∧ tp2 ≟ 6 ∨ tp1 ≟ 2 ∧ tp2 ≟ 1 ∨ tp1 ≟ 4 ∧ tp2 ≟ 1 ∨ tp1 ≟ 6 ∧ tp2 ≟ 1 ∨ tp1 ≟ 6 ∧ tp2 ≟ 2 ∨ tp1 ≟ 6 ∧ tp2 ≟ 4, ¬indFa5, ¬indFa6]]]], Si[tp1 ≟ 1 ∧ tp2 ≟ 5 ∨ tp1 ≟ 3 ∧ tp2 ≟ 6 ∨ tp1 ≟ 5 ∧ tp2 ≟ 6 ∨ tp1 ≟ 6 ∧ tp2 ≟ 5, indFa2, Si[tp1 ≟ 2 ∧ tp2 ≟ 6 ∨ tp1 ≟ 4 ∧ tp2 ≟ 6, indFa3, Si[tp1 ≟ 1 ∧ tp2 ≟ 3 ∨ tp1 ≟ 3 ∧ tp2 ≟ 1 ∨ tp1 ≟ 5 ∧ tp2 ≟ 1 ∨ tp1 ≟ 6 ∧ tp2 ≟ 3, indFa4, Si[tp1 ≟ 1 ∧ tp2 ≟ 2 ∨ tp1 ≟ 1 ∧ tp2 ≟ 4 ∨ tp1 ≟ 1 ∧ tp2 ≟ 6 ∨ tp1 ≟ 2 ∧ tp2 ≟ 1 ∨ tp1 ≟ 4 ∧ tp2 ≟ 1 ∨ tp1 ≟ 6 ∧ tp2 ≟ 1 ∨ tp1 ≟ 6 ∧ tp2 ≟ 2 ∨ tp1 ≟ 6 ∧ tp2 ≟ 4, indFa5, indFa6]]]]]

Etablissons un indicateur de visibilité pour le plan contenant mOr’, mOy’, mOx et mOz’.

On aura donc (pour la face arrière lorsque le cube mobile est dans sa position de départ),

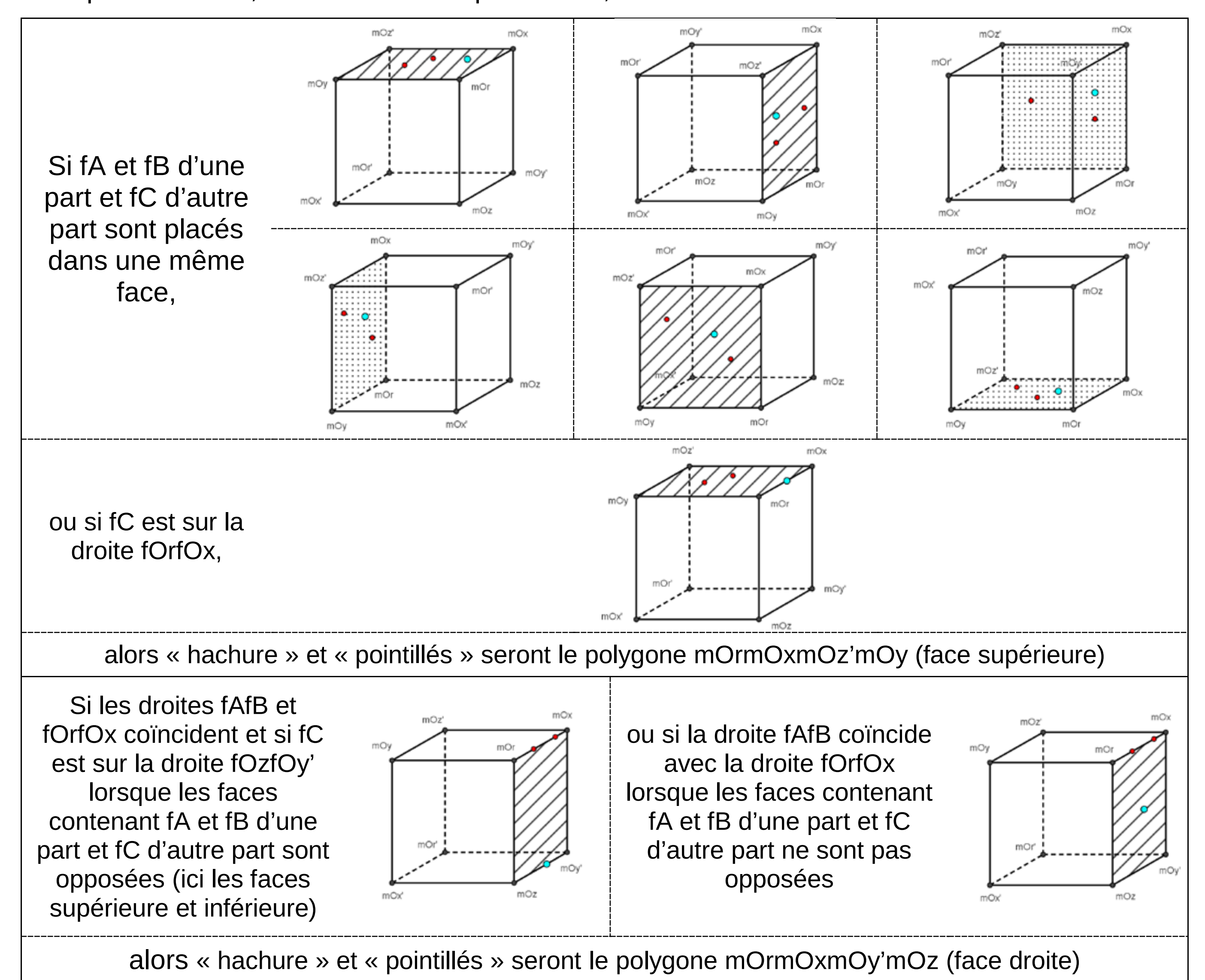
69) Saisie : indbsarrière = ¬indbsavant

Il existe deux sortes de section

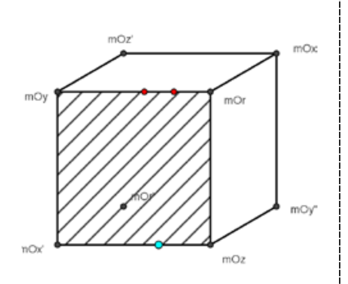
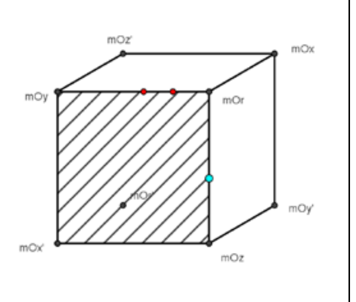
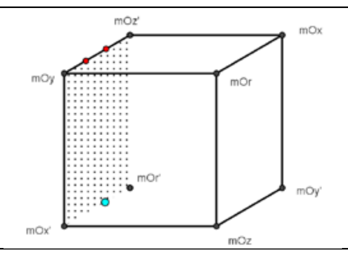
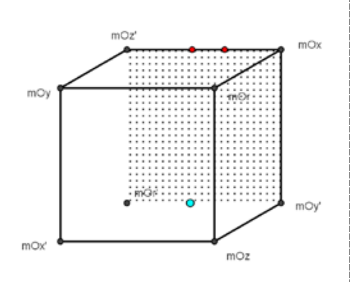
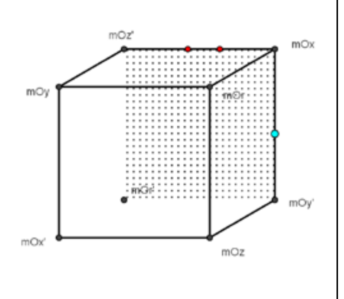
Lorsque fA, fB et fC sont distincts deux à deux et non alignés, deux types de sections peuvent être trouvés. Une section coïncidant avec une face devra être hachurée ou pointillée selon que cette face est vue ou cachée, et une section disjointe de toute face devra être grisée.

Nous créerons les quadrilatères « hachure » et « pointillés » en raisonnant comme suit.

Dans les dessins suivants, le cube mobile est dans sa position de départ. Par conséquent, dans cette position, « fOr » coïncide avec « mOr », « fOx » coïncide avec « mOx » et cetera.
Pour plus de clarté, nous avons masqué « fOr », « fOx » et cetera.

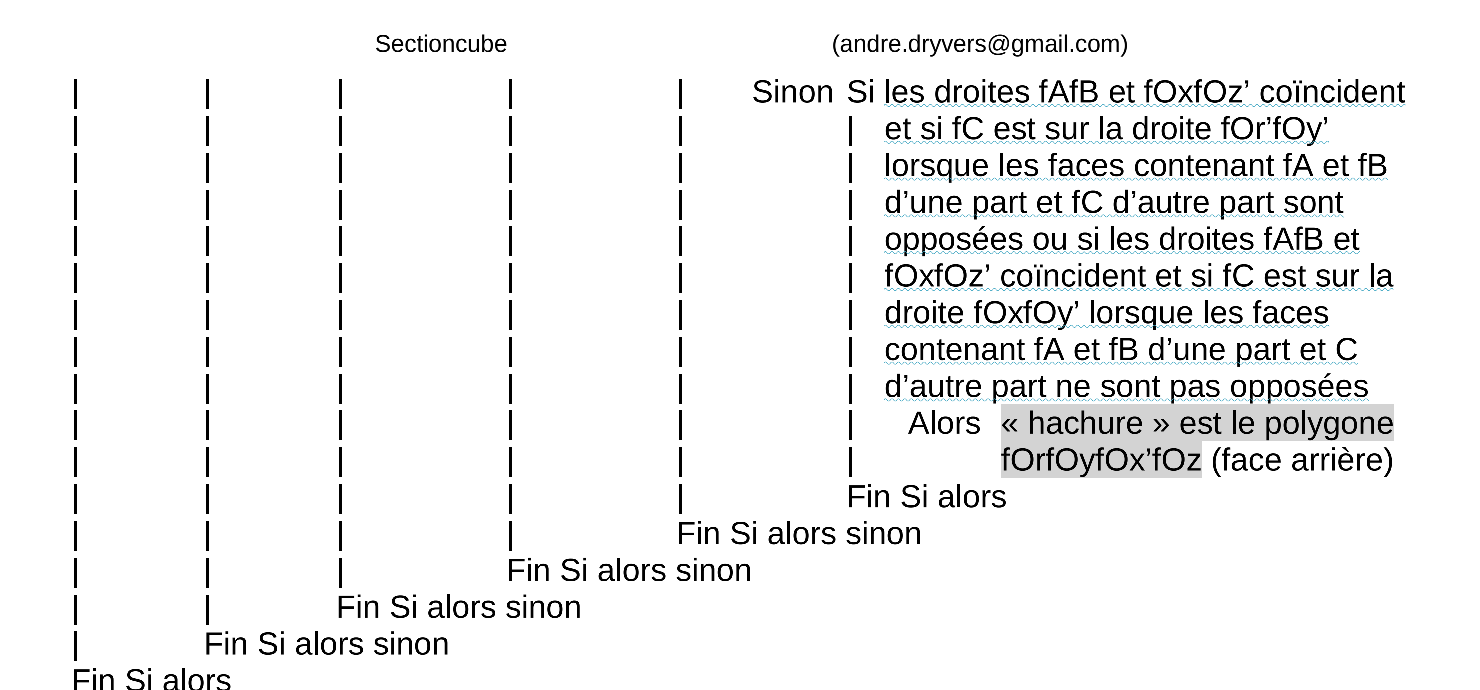


Si fA et fB d’une part et fC d’autre part sont placés dans une même face,		
ou si fC est sur la droite fOrfOx,		
alors « hachure » et « pointillés » seront le polygone mOrmOxmOz’mOy (face supérieure)		
Si les droites fAfB et fOrfOx coïncident et si fC est sur la droite fOzfOy’ lorsque les faces contenant fA et fB d’une part et fC d’autre part sont opposées (ici les faces supérieure et inférieure)		ou si la droite fAfB coïncide avec la droite fOrfOx lorsque les faces contenant fA et fB d’une part et fC d’autre part ne sont pas opposées
alors « hachure » et « pointillés » seront le polygone mOrmOxmOy’mOz (face droite)		

<table>
<tr><td>Si les droites fAfB et fOrfOx coïncident et si fC est sur la droite fOzfOx' lorsque les faces contenant fA et fB d'une part et fC d'autre part sont opposées (ici les faces supérieure et inférieure)
</td><td>ou si les droites fAfB et fOrfOx coïncident et si fC est sur la droite fOrfOz lorsque les faces contenant fA et fB d'une part et fC d'autre part ne sont pas opposées
</td></tr>
<tr><td colspan="2">alors « hachure » et « pointillés » seront le polygone mOrmOymOx'mOz (face avant)</td></tr>
<tr><td colspan="2">Si les droites fAfB et fOyfOz' coïncident et si fC est sur la droite fOx'fOr',
alors « hachure » et « pointillés » seront le polygone mOymOx'mOr'mOz' (ici la face gauche)
</td></tr>
<tr><td>Si les droites fAfB et fOxfOz' coïncident et si fC est sur la droite fOr'fOy' lorsque les faces contenant fA et fB d'une part et fC d'autre part sont opposées (ici les faces supérieure et inférieure)
</td><td>ou si les droites fAfB et fOxfOz' coïncident et si fC est sur la droite fOxfOy' lorsque les faces contenant fA et fB d'une part et fC d'autre part ne sont pas opposées
</td></tr>
<tr><td colspan="2">alors « hachure » et « pointillés » seront le polygone mOrmOymOx'mOz (face arrière)</td></tr>
</table>

Voici un schéma expliquant la construction de « hachure »

```
Si fA, fB et fC sont distincts deux à deux et ne sont pas alignés
|  Alors  Si fA, fB et fC sont dans une même face ou si fOr, fC et fOx sont alignés
|         |  Alors hachure est le polygone fOrfOxfOz'fOy (face supérieure)
|         |  Sinon  Si les droites fAfB et fOrfOx coïncident et si fC est sur la droite fOzfOy'
|         |         |  lorsque les faces contenant fA et fB d'une part et fC d'autre part sont
|         |         |  opposées ou si la droite fAfB coïncide avec la droite fOrfOx lorsque
|         |         |  les faces contenant fA et fB d'une part et fC d'autre part ne sont pas
|         |         |  opposées
|         |         |     Alors « hachure » est le polygone fOrfOxfOy'fOz (face droite)
|         |         |     Sinon  Si les droites fAfB et fOrfOx coïncident et si fC est sur la droite
|         |         |            |  fOzfOx' lorsque les faces contenant fA et fB d'une part et fC
|         |         |            |  d'autre part sont opposées ou si les droites fAfB et fOxfOz'
|         |         |            |  coïncident et si fC est sur la droite fOxfOy' lorsque les faces
|         |         |            |  contenant fA et fB d'une part et fC d'autre part ne sont pas
|         |         |            |  opposées
|         |         |            |     Alors « hachure » est le polygone fOrfOyfOx'fOz (face
|         |         |            |                                                      avant)
|         |         |            |     Sinon  Si les droites fAfB et fOyfOz' coïncident et si fC est
|         |         |            |            |  sur la droite fOx'fOr'
|         |         |            |            |     Alors  « hachure » est le polygone
|         |         |            |            |            fOyfOx'fOr'fOz' (face gauche)
```



```
|    |    |    |    |    Sinon  Si les droites fAfB et fOxfOz’ coïncident
|    |    |    |    |         |  et si fC est sur la droite fOr’fOy’
|    |    |    |    |         |  lorsque les faces contenant fA et fB
|    |    |    |    |         |  d’une part et fC d’autre part sont
|    |    |    |    |         |  opposées ou si les droites fAfB et
|    |    |    |    |         |  fOxfOz’ coïncident et si fC est sur la
|    |    |    |    |         |  droite fOxfOy’ lorsque les faces
|    |    |    |    |         |  contenant fA et fB d’une part et C
|    |    |    |    |         |  d’autre part ne sont pas opposées
|    |    |    |    |         |    Alors  « hachure » est le polygone
|    |    |    |    |         |           fOrfOyfOx’fOz (face arrière)
|    |    |    |    |         Fin Si alors
|    |    |    |    Fin Si alors sinon
|    |    |    Fin Si alors sinon
|    |    Fin Si alors sinon
|    Fin Si alors sinon
Fin Si alors
```

Si[ind3pdis ∧ (¬ind3pali), Si[tp1 ≟ tp2 ∨ pointsurdroite[fC, Droite[fOr, fOx]], Polygone[fOr, fOx, fOz', fOy], Si[Droite[fA, fB] ≟ Droite[fOr, fOx] ∧ Si[indfopo, pointsurdroite[fC, Droite[fOz, fOy']], true], Polygone[fOr, fOx, fOy', fOz], Si[Droite[fA, fB] ≟ Droite[fOr, fOy] ∧ Si[indfopo, pointsurdroite[fC, Droite[fOx', fOz]], pointsurdroite[fC, Droite[fOr, fOz]]], Polygone[fOr, fOy, fOx', fOz], Si[Droite[fA, fB] ≟ Droite[fOy, fOz'] ∧ pointsurdroite[fC, Droite[fOx', fOr']], Polygone[fOy, fOz', fOr', fOx'], Si[Droite[fA, fB] ≟ Droite[fOz', fOx] ∧ Si[indfopo, pointsurdroite[fC, Droite[fOr', fOy']], pointsurdroite[fC, Droite[fOx, fOy']]], Polygone[fOx, fOy', fOr', fOz']]]]]]

70) hachure = Si[ind3pdis ∧ (¬ind3pali), Si[tp1 ≟ tp2 ∨ pointsurdroite[fC, Droite[fOr, fOx]], Polygone[fOr, fOx, fOz', fOy], Si[Droite[fA, fB] ≟ Droite[fOr, fOx] ∧ Si[indfopo, pointsurdroite[fC, Droite[fOz, fOy']], true], Polygone[fOr, fOx, fOy', fOz], Si[Droite[fA, fB] ≟ Droite[fOr, fOy] ∧ Si[indfopo, pointsurdroite[fC, Droite[fOx', fOz]], pointsurdroite[fC, Droite[fOr, fOz]]], Polygone[fOr, fOy, fOx', fOz], Si[Droite[fA, fB] ≟ Droite[fOy, fOz'] ∧ pointsurdroite[fC, Droite[fOx', fOr']], Polygone[fOy, fOz', fOr', fOx'], Si[Droite[fA, fB] ≟ Droite[fOz', fOx] ∧ Si[indfopo, pointsurdroite[fC, Droite[fOr', fOy']], pointsurdroite[fC, Droite[fOx, fOy']]], Polygone[fOx, fOy', fOr', fOz']]]]]]

Créons un indicateur pour afficher ou non le polygone « hachure »

Il suffit reprendre la formule donnant « hachure » et d’y remplacer les plans sur lesquels « hachure » se pose par les indicateurs de ces plans.

71) indhachure = Si[ind3pdis ∧ (¬ind3pali), Si[tp1 ≟ tp2 ∨ pointsurdroite[fC, Droite[fOr, fOx]], indbssupérieure, Si[Droite[fA, fB] ≟ Droite[fOr, fOx] ∧ Si[indfopo, pointsurdroite[fC, Droite[fOz, fOy']], true], indbsdroite, Si[Droite[fA, fB] ≟ Droite[fOr, fOy] ∧ Si[indfopo, pointsurdroite[fC, Droite[fOx', fOz]], pointsurdroite[fC, Droite[fOr, fOz]]], indbsavant, Si[Droite[fA, fB] ≟ Droite[fOy, fOz'] ∧ pointsurdroite[fC, Droite[fOx', fOr']], indbsgauche, Si[Droite[fA, fB] ≟ Droite[fOz', fOx] ∧ Si[indfopo, pointsurdroite[fC, Droite[fOr', fOy']], pointsurdroite[fC, Droite[fOx, fOy']]], indbsarrière]]]]]

Nous affecterons « hachure » des propriétés suivantes ; Propriétés, onglet Style, Epaisseur du trait : 3, Remplir : Hachures, Angle : 45°, Espacement : 15, Condition pour afficher l’objet : (¬ppoints) ∧ indhachure ∧ vuesection.

Nous créerons également le polygone « pointillés » qui remplacera « hachure » lorsque ce dernier sera dans une face cachée. Le polygone « pointillé » coïncidera donc avec le polygone « hachure », la seule différence entre ceux-ci étant leur aspect hachuré pour une face vue et pointillés pour une face cachée.

72) Saisie : pointillés = Si[¬indhachure, hachure] (Propriétés, onglet Style, Epaisseur du trait : 0, Remplir : Points, Espacement : 5, Onglet Avancé, Condition pour afficher l'objet : (¬ppoints) ∧ (¬indhachure) ∧ vuesection)

Dans les propriétés des cases à cocher « sommets sections » et « bords de section », onglet Avancé, Condition pour afficher l'objet, nous écrirons ¬EstDéfini[hachure]. Ainsi, ces boites à cocher disparaîtront lorsque la section coïncidera avec une face.

Créerons le quadrilatère « section ».

Définissons le booléen « sq» (section quelconque) qui nous permettra de ne réaliser les constructions suivantes seulement en dehors des cas particuliers déjà décrits.

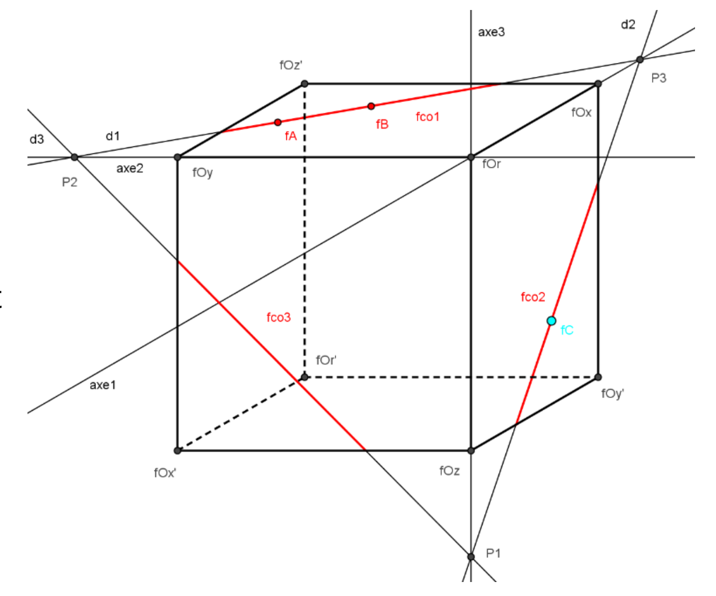
« sq » prendra la valeur true lorsque l'on aura la conjonction des conditions suivantes :

« hachure » ne sera pas défini, « fA », « fB » et « fC » seront disjoints deux à deux et enfin « fA », « fB » et « fC » ne seront pas alignés.

73) Saisie : sq = (¬EstDéfini[hachure]) ∧ ind3pdis ∧ (¬ind3pali)

Considérerons quelques cas de figures.

Observons la construction de la section d'un plan dans le cube dans le cas où fA et fB sont placés dans le plan de la face supérieure tandis que fC est placé dans le plan de la face latérale droite.
« d1 », la droite déterminée par fA et fB coupe « axe1 », la droite déterminé par fOr et fOx, en P3.



74) Droite : d1 = Si[sq, Droite[fA, fB]]

75) Droite : axe1 = Si[sq, Droite[fOr, fOx]]

76) Point : P3 = Intersection[d1, axe1]

Quelques observations provisoires !

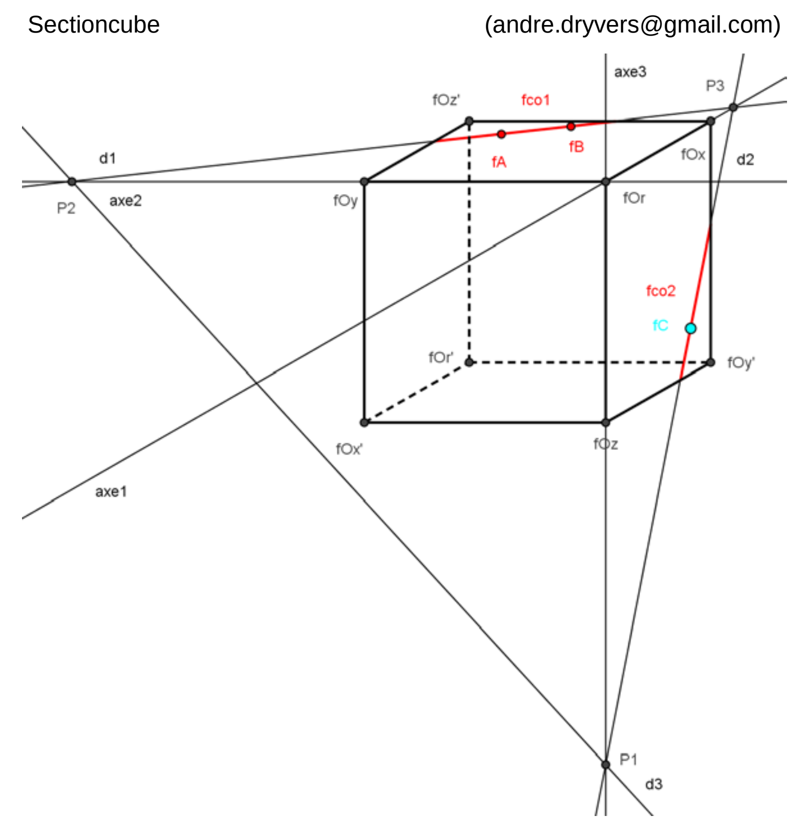
d1 coupe « axe2 », la droite déterminée provisoirement par fOr et fOy, en « P2 ».

« d2 », la droite d2 déterminée provisoirement par P3 et fC, coupe « axe3 », la droite déterminée provisoirement par fOr et fOz, en « P1 ».

La droite « d3 » sera provisoirement déterminée par P1 et P2.

Dans cette configuration, nous possédons déjà les six points permettant de déterminer les sommets de la section. En effet, d1 coupe la face supérieure en y traçant « fco1 », d2 coupe la face latérale droite en y traçant « fco2 » et d3 coupe la face avant en y traçant « fco3 », les extrémités de ces trois segments étant les six points évoqués.

Mais on ne trouvera pas toujours aussi facilement tous ces points. Ainsi, dans la figure suivante, ne sont déterminés que « fco1 » et « fco2 ». Les extrémités de ces deux segments ne donnent que quatre points pour construire les sommets de la section.



Mais nous savons que les intersections de deux plans parallèles avec un troisième plan sécant à ceux-ci sont deux droites parallèles.

Soit « axe1' » la droite déterminée par « fOz » et « fOy' ».

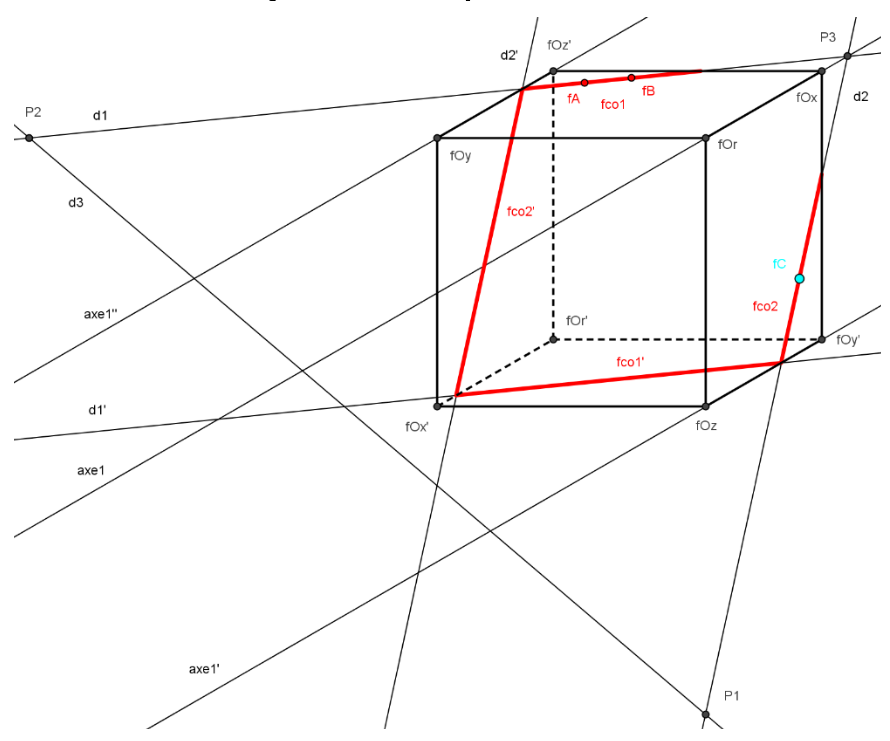
77) Droite : axe1' = Si[sq, Droite[fOz, fOy']]

Remarquons alors que « d1' », la parallèle à d1 passant par l'intersection de d2 avec la droite « axe1' », coupera la face inférieure en traçant « fco1' ».

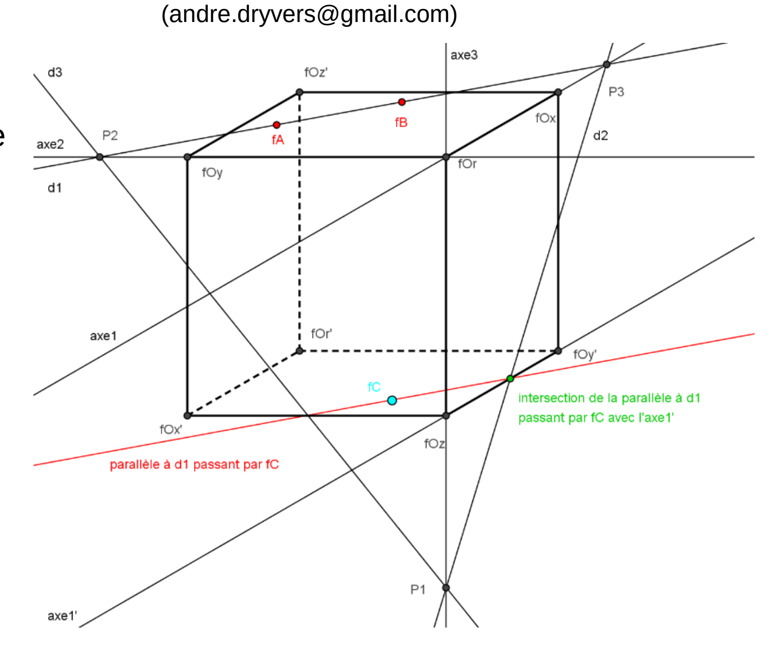
Soit « axe1'' » la droite déterminée par « fOy » et « fOz' ».

78) Droite : axe1'' = Droite[fOy, fOz']

Remarquons alors que « d2' », la parallèle à d2 passant par l'intersection de d1 avec la droite « axe1'' », coupera la face latérale gauche en traçant « fco2' ».



Observons maintenant la construction de la section d'un plan dans le cube dans le cas où fA et fB sont placés dans le plan de la face supérieure tandis que fC est placé dans le plan de la face inférieure.



Cette fois, la droite d2 ne sera plus déterminée par P3 et fC mais bien par P3 et le point d'intersection de la parallèle à d1 passant par fC avec l'axe1'.

79) d2 = Si[indfopo, Droite[P3, Intersection[Droite[fC, d1], axe1']], Droite[P3, fC]]

80) d2' = Droite[Intersection[d1, axe1"], d2]

Observons maintenant la construction de la section d'un plan dans le cube dans le cas où, fA et fB étant placés dans le plan de la face supérieure tandis que fC est placé dans le plan de la face inférieure, la droite d1 est parallèle à l'axe1.

Remarquons que dans ce cas, « P3 » est rejeté à l'infini et que « d2 » n'existera plus. Il faudra donc définir « d1' » comme la parallèle à « d1 » passant par C.

81) d1' = Si[indfopo, Droite[fC, d1], Droite[Intersection[d2, axe1'], d1]]

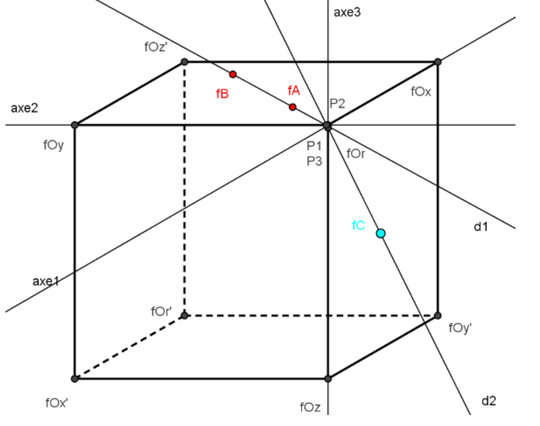
Remarquons également que dans ce cas, « P1 » n'existe plus. Soit « axe2' » la droite déterminée provisoirement par fOx' et fOz. Pour déterminer la section, il faudra donc définir provisoirement « d3 » comme la droite déterminée par « P2 » et l'intersection de « d1' » avec « axe2' » tandis que « d3' » sera provisoirement l'image de « d3 » par la translation appliquant « fOr » sur « fOx ».

<u>Il est parfois nécessaire de changer la position des axes</u>

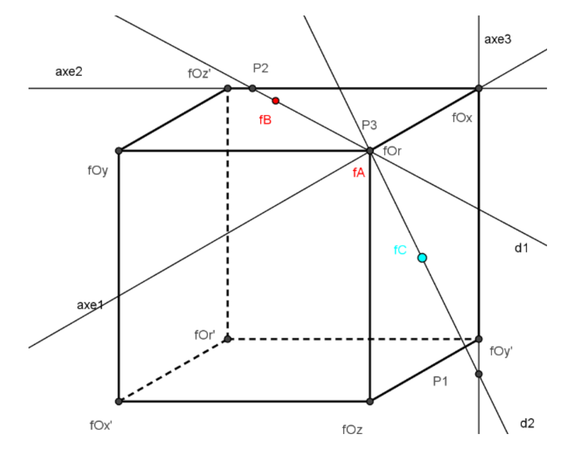
Si les points fA et fB sont placés de telle manière que « d1 » passe par l'intersection de « axe1 » et « axe2 », les points « P1 », « P2 » et « P3 » se confondent et par conséquent, les droites « d2 » et « d3 » ne peuvent plus être déterminées.

Cette situation peut se déceler par le fait que « P3 » et « fOr » sont confondus.

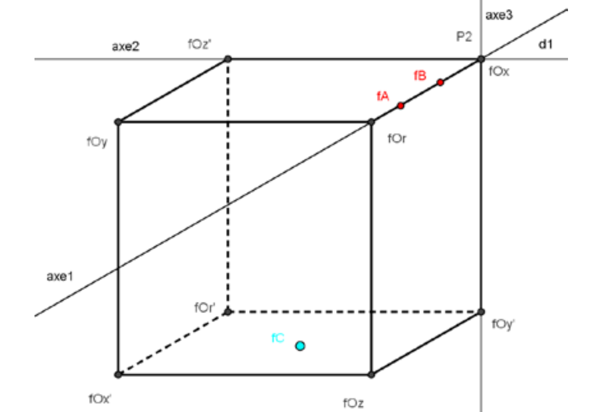
Pour pallier cette indétermination, nous déciderons que, dans ce cas, l'axe2 sera la droite passant par fOx et fOz' et nous déciderons que l'axe3 sera la droite passant par fOx et fOy'.



Nous obtiendrons alors des points P1, P2 et P3 distincts deux à deux.



Si les points fA et fB sont placés de telle manière que « d1 » est parallèle à « axe1 », « P3 » est rejeté à l'infini et est donc distinct de « fOr ».



Si « d1 » est confondue avec « axe1 », un changement des axes est nécessaire mais « P3 » n'existant plus, il devient impossible de déceler la nécessité de ce changement de repère par la superposition de « P3 » et « fOr».

Nous concevrons donc un indicateur pour déceler ce cas.

Ouvrons l'outil « droitesparallèles.ggt » et créons l'indicateur « indrep » pour déceler la nécessité d'un changement de position des axes.

Voici un schéma montrant le fonctionnement de cet indicateur.

Si P3 existe
| Alors la valeur de indrep sera True ou False selon que P3 et fOr sont ou non confondus
| Sinon Si d1 et axe1 sont confondues
| | alors la valeur de indrep sera « True »
| | sinon la valeur de indrep sera « False »
| Fin si alors sinon
Fin si alors sinon

82) Saisie : indrep = Si[EstDéfini[P3], P3 $\stackrel{?}{=}$ fOr, Si[d1 $\stackrel{?}{=}$ axe1, true, false]]

Soit « axe2 » la droite passant, hors cas particuliers, par fOz' et fOx ou par fOr et fOy selon la valeur de « indrep » tandis que « axed2' » sera la droite passant, hors cas particuliers, par fOr' et fOy' ou par fOx' et fOz selon la valeur de « indrep »

83) Saisie : axe2 = Si[sq, Si[indrep, Droite[fOz', fOx], Droite[fOr, fOy]]]

84) Saisie : axe2' = Si[sq, Si[indrep, Droite[fOr', fOy'], Droite[fOx', fOz]]]

Soit « axe » la droite passant, hors cas particuliers, par fOr et Oy' ou la droite passant par fOr et fOz selon la valeur de « indrep ».

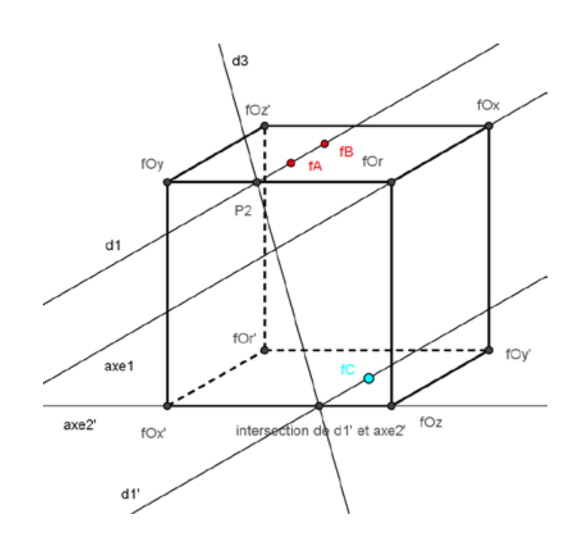
85) Saisie : axe3 = Si[sq, Si[indrep, Droite[fOx, fOy'], Droite[fOr, fOz]]]

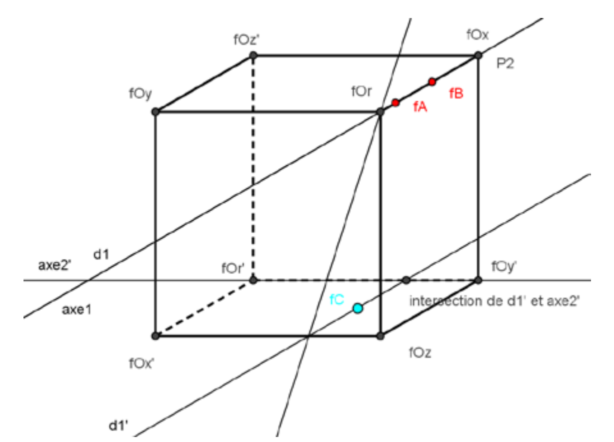
86) Point : P1 = Intersection[d2, axe3]

87) Point : P2 = Intersection[d1, axe2]

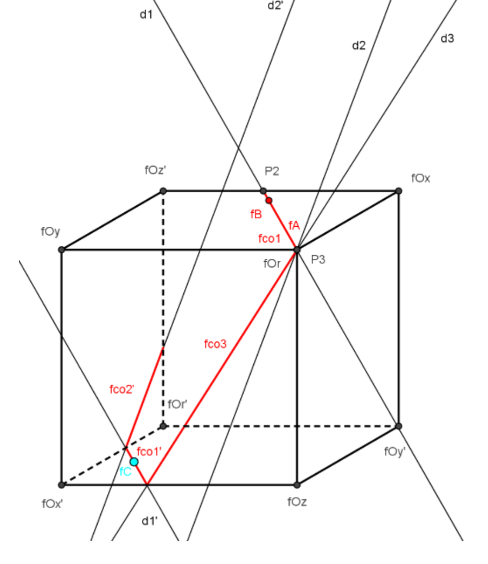
Construisons d3

Lorsque d1 est parallèle à l'axe1 sans être confondue avec lui et que les faces contenant fA et fB d'une part et fC d'autre part sont opposées, d3 doit être la droite passant par P2 et par le point d'intersection de d1' et de l'axe2'.

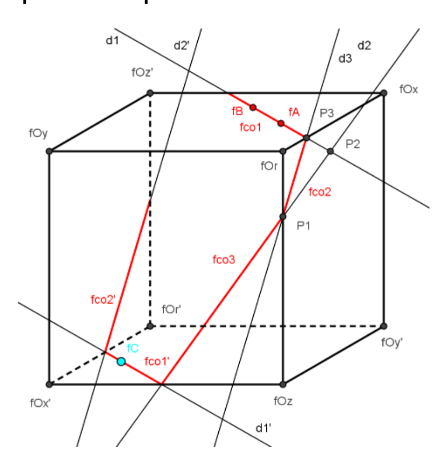




Lorsque d1 est confondue avec l'axe1 et que les faces contenant fA et fB d'une part et fC d'autre part sont opposées, d3 doit être l'image de la droite passant par P2 et par le point d'intersection de d1' et de l'axe2' par la translation qui applique fOx sur fOr.



Lorsque, d1 n'étant pas parallèle à l'axe1, P3 est confondu avec fOr (l'indicateur indrep prend la valeur true), d3 doit être la parallèle à la droite déterminée par P1 et P2 passant par P3.



Lorsque, d1 n'étant pas parallèle à l'axe1, P3 n'est pas confondu avec fOr (l'indicateur indrep prend la valeur false), d3 doit être la droite déterminée par P1 et P2.

Voici un schéma montrant la construction de d3.

Si les faces contenant fA et fB d'une part et fC d'autre part sont
| opposées et que d1 et l'axe1 sont parallèles
| | Alors Si d1 passe par fOr
| | | Alors d3 est l'image de la droite passant par P2 et l'intersection de d1' avec l'axe2' par la translation qui applique fOx sur fOr
| | | Sinon d3 est la droite passant par P2 et l'intersection de d1' avec l'axe2'
| | Fin si alors sinon
| | Sinon Si d1 passe par fOr
| | | Alors d3 est la parallèle à la droite P1P2 passant par P3,
| | | Sinon d3 est la droite passant par P1 et P2
| | Fin si alors sinon
| Fin si alors sinon
Fin si alors sinon

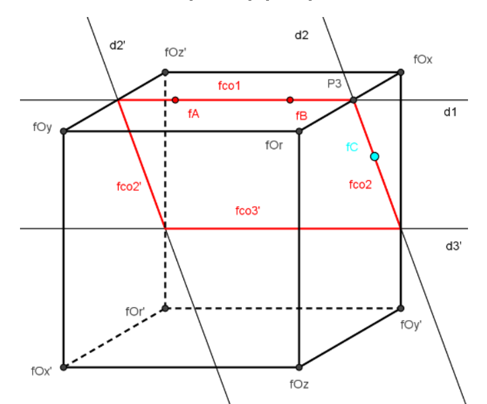
En Geogebra, cela donne

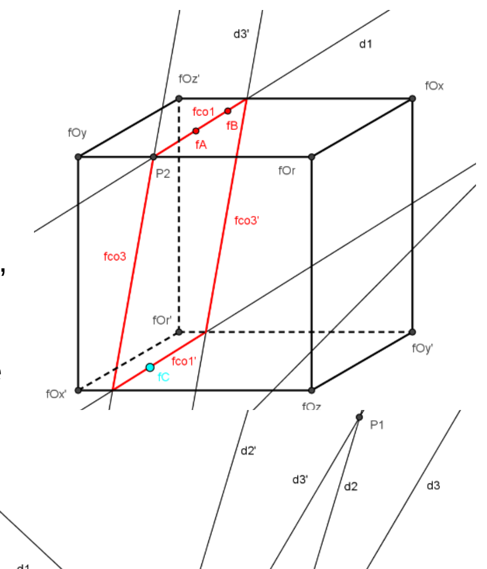
d3 = Si[indfopo ∧ droitesparallèles[fA, d1, fOr, axe1], Si[indrep, Translation[Droite[P2, Intersection[d1', axe2']], Vecteur[fOx, fOr]], Droite[P2, Intersection[d1', axe2']]], Si[indrep, Droite[P3, Droite[P1, P2]], Droite[P1, P2]]]

88) d3 = Si[indfopo ∧ droitesparallèles[fA, d1, fOr, axe1], Si[indrep, Translation[Droite[P2, Intersection[d1', axe2']], Vecteur[fOx, fOr]], Droite[P2, Intersection[d1', axe2']]], Si[indrep, Droite[P3, Droite[P1, P2]], Droite[P1, P2]]]

Construisons d3'

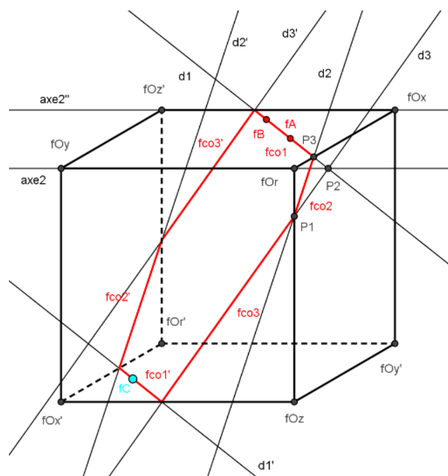
Lorsque d1 est parallèle à l'axe1, d3' est l'image de d3 par la translation qui applique fOx sur fOr.

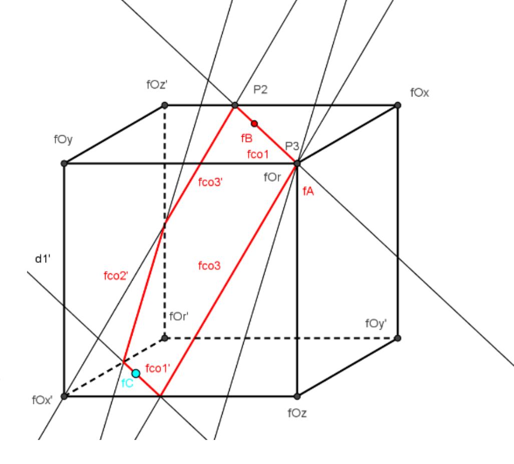




Lorsque d1 est parallèle à la droite passant par fOr et fOy, d3' est la parallèle à d1 passant par l'intersection de d2 et de la droite déterminée par fOx et fOy'.

Lorsque, d1 n'étant pas parallèle ni à l'axe1 ni à l'axe2, P3 est confondu avec fOr (l'indicateur indrep prend la valeur true), d3' doit être la droite déterminée par P1 et P2.





Lorsque, d1 n'étant pas parallèle ni à l'axe1 ni à l'axe2, P3 n'est pas confondu avec fOr (l'indicateur indrep prend la valeur false), d3' doit être la parallèle à la droite déterminée par P1 et P2 passant par l'intersection de d1 et de l'axe2''.

Voici un schéma montrant la construction de d3'.

Si d1 et l'axe1 sont parallèles
| | Alors d3' est l'image de d3 par la translation qui applique fOx sur fOr
| | Sinon Si d1 et l'axe2 sont parallèles
| | | Alors d3' est la parallèle à d1 passant par l'intersection de d2 et de la
| | | droite déterminée par fOx et fOy'
| | | Sinon Si d1 passe par fOr
| | | | Alors d3' est la droite déterminée par P1 et P2
| | | | Sinon d3' est la parallèle à la droite déterminée par P1 et P2
| | | | passant par l'intersection de d1 et de l'axe2''
| | | Fin si alors sinon
| | Fin si alors sinon
| Fin si alors sinon
Fin si alors sinon

En Geogebra, cela donne

d3 = Si[droitesparallèles[fA, d1, fOr, axe1], Translation[d3, Vecteur[fOr, fOx]], Si[droitesparallèles[fA, d1, fOr, Droite[fOr, fOy]], Droite[Intersection[d2, Droite[fOx, fOy']], d1], Si[indrep, Droite[P1, P2], Droite[Intersection[d1, axe2''], Droite[P1, P2]]]]]

89) d3' = Si[droitesparallèles[fA, d1, fOr, axe1], Translation[d3, Vecteur[fOr, fOx]], Si[droitesparallèles[fA, d1, fOr, Droite[fOr, fOy]], Droite[Intersection[d2, Droite[fOx, fOy']], d1], Si[indrep, Droite[P1, P2], Droite[Intersection[d1, axe2''], Droite[P1, P2]]]]]

Déterminons les traces des droites d1, d1', d2, d2', d3 et d3' dans les faces du cube.

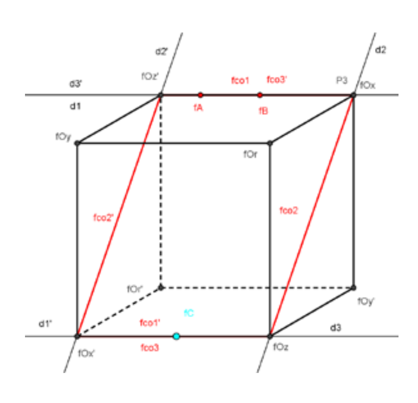
Elles seront obtenues avec l'outil « tracedansface.ggt » décris dans les pages précédentes. Ouvrons donc cet outil.

90) Outil trace dans face : fco1 = tracedansface[d1, fOr, fOx, fOz', fOy] (face supérieure)

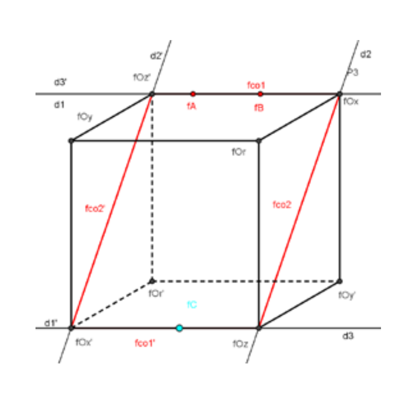
91) Outil trace dans face : fco1' = tracedansface[d1', fOr', fOx', fOz, fOy'] (face inférieure)

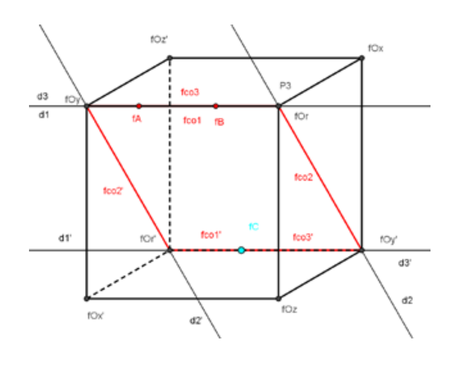
92) Outil trace dans face : fco2 = tracedansface[d2, fOr, fOx, fOy', fOz] (face droite)

93) Outil trace dans face : fco2' = tracedansface[d2', fOy, fOz', fOr', fOx'] (face gauche)

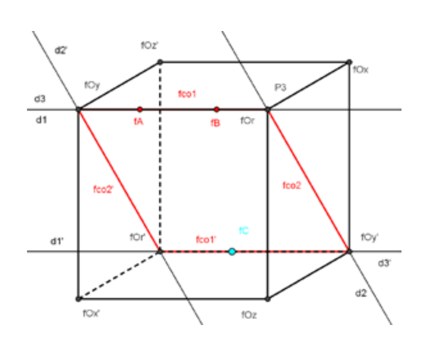


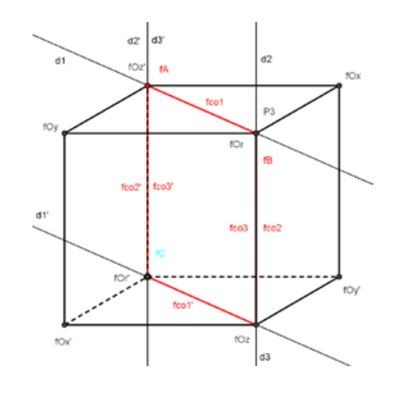
Lorsque fA et fB sont placés comme ci-contre, d3 et d1' coïncident de même que d3' et d1. Pour ne pas obtenir des côtés « fco3 » et « fco1' » superposés de même que des côtés « fco3' » et « fco1 » superposés, nous empêcherons l'existence de « fco3 » lorsque d3 et d1' seront confondues et celle de « fco3' » lorsque d3' et d1 seront confondues.



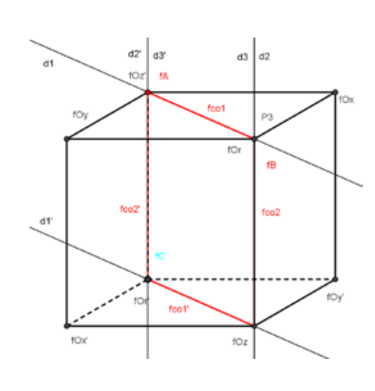


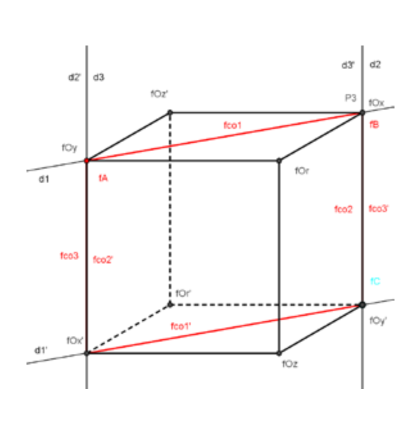
Lorsque fA et fB sont placés comme ci-contre, d3 et d1 coïncident de même que d3' et d1'. Pour ne pas obtenir des côtés « fco3 » et « fco1 » superposés de même que des côtés « fco3' » et « fco1' » superposés, nous empêcherons l'existence de « fco3 » lorsque d3 et d1 seront confondues et celle de « fco3' » lorsque d3' et d1' seront confondues.



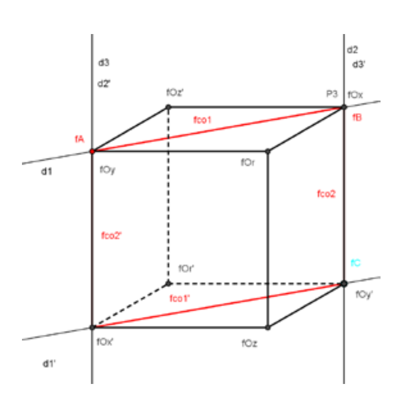


Lorsque fA et fB sont placés comme ci-contre, d3 et d2 coïncident de même que d3' et d2'. Pour ne pas obtenir des côtés « fco3 » et « fco2 » superposés de même que des côtés « fco3' » et « fco2' » superposés, nous empêcherons l'existence de « fco3 » lorsque d3 et d2 seront confondues et celle de « fco3' » lorsque d3' et d2' seront confondues.





Lorsque fA et fB sont placés comme ci-contre, d3 et d2' coïncident de même que d3' et d2. Pour ne pas obtenir des côtés « fco3 » et « fco2' » superposés de même que des côtés « fco3' » et « fco2 » superposés, nous empêcherons l'existence de « fco3 » lorsque d3 et d2' seront confondues et celle de « fco3' » lorsque d3' et d2 seront confondues.



94) Saisie : fco3 = Si[(d3 ≠ d1) ∧ (d3 ≠ d1') ∧ (d3 ≠ d2) ∧ (d3 ≠ d2'), tracedansface[d3, fOr, fOy, fOx', fOz]] (face avant)

95) Saisie : fco3' = Si[(d3' ≠ d1) ∧ (d3' ≠ d1') ∧ (d3' ≠ d2) ∧ (d3' ≠ d2'), tracedansface[d3', fOx, fOy', fOr', fOz']] (face arrière)

Nous masquerons tous les points et droites utilisés pour nos constructions.

<u>Déterminons les points qui permettront d'obtenir les sommets de la section</u>.

fSS1, fSS2, fSS3 et fSS4, placés sur les arêtes de la face supérieure, seront les intersections des segments fco1 et fco3', fco1 et fco2', fco1 et fco3, fco1 et fco2.

fSS5, fSS6, fSS7 et fSS8, placés sur les arêtes latérales, seront les intersections de fco3' et fco2, fco3' et fco2', fco3 et fco2', fco3 et fco2.

fSS9, fSS10, fSS11 et fSS12, placés sur les arêtes de la face inférieure, seront les intersections de fco1' et fco3', fco1' et fco2', fco1' et fco3, fco1' et fco2.

96) fSS1 = Intersection[fco1, fco3']

97) fSS2 = Intersection[fco1, fco2']

98) fSS3 = Intersection[fco1, fco3]

99) fSS4 = Intersection[fco1, fco2]

100) fSS5 = Intersection[fco2, fco3']

101) fSS6 = Intersection[fco2', fco3']

102) fSS7 = Intersection[fco3, fco2']

103) fSS8 = Intersection[fco2, fco3]

104) fSS9 = Intersection[fco1', fco3']

105) fSS10 = Intersection[fco1', fco2']

106) fSS11 = Intersection[fco3, fco1']

107) fSS12 = Intersection[fco2, fco1']

<u>Reportons les éléments utilisés pour les constructions du cube fixe dans le cube mobile</u>

Puisque fSS1, fSS2, fSS3 et fSS4 sont situés dans le quadrilatère OrOxOyOz', mSS1, mSS2, mSS3 et mSS4 seront situés dans le quadrilatère mOrmOxmOymOz'.

mSS1 sera le sommet de la section placé sur l'arête arrière de la face supérieure du cube mobile dans sa position de départ.

108) Saisie : mSS1 = repères2dsynchronisés[fSS1, fOr, fOx, fOy, mOr, mOx, mOy]

mSS2 sera le sommet de la section placé sur l'arête gauche de la face supérieure du cube mobile dans sa position de départ.

109) Saisie : mSS2 = repères2dsynchronisés[fSS2, fOr, fOx, fOy, mOr, mOx, mOy]

mSS3 sera le sommet de la section placé sur l’arête avant de la face supérieure du cube mobile dans sa position de départ.

110) Saisie : mSS3 = repères2dsynchronisés[fSS3, fOr, fOx, fOy, mOr, mOx, mOy]

mSS4 sera le sommet de la section placé sur l’arête droite de la face supérieure du cube mobile dans sa position de départ.

111) Saisie : mSS4 = repères2dsynchronisés[fSS4, fOr, fOx, fOy, mOr, mOx, mOy]

Puisque fSS5 et fSS6 sont situés dans le quadrilatère OrOxOy’Oz, mSS5 et mSS6 seront situés dans le quadrilatère mOrmOxmOy’mOz.

mSS5 sera le sommet de la section placé sur l’arête droite de la face arrière du cube mobile dans sa position de départ.

112) Saisie : mSS5 = repères2dsynchronisés[fSS5, fOr', fOy', fOz', mOr', mOy', mOz']

mSS6 sera le sommet de la section placé sur l’arête gauche de la face arrière du cube mobile dans sa position de départ.

113) Saisie : mSS6 = repères2dsynchronisés[fSS6, fOr', fOy', fOz', mOr', mOy', mOz']

Puisque fSS7 et fSS8 sont situés dans le quadrilatère OrOyOx’Oz, mSS7 et mSS8 seront situés dans le quadrilatère mOrmOymOx’mOz.

mSS7 sera le sommet de la section placé sur l’arête gauche de la face avant du cube mobile dans sa position de départ.

114) Saisie : mSS7 = repères2dsynchronisés[fSS7, fOz, fOx', fOr, mOz, mOx', mOr]

mSS8 sera le sommet de la section placé sur l’arête droite de la face avant du cube mobile dans sa position de départ.

115) Saisie : mSS8 = repères2dsynchronisés[fSS8, fOz, fOx', fOr, mOz, mOx', mOr]

Puisque fSS9, fSS10, fSS11 et fSS12 sont situés dans le quadrilatère Or’Ox’Oy’Oz, mSS1, mSS2, mSS3 et mSS4 seront situés dans le quadrilatère mOr’mOx’mOy’mOz.

mSS9 sera le sommet de la section placé sur l’arête arrière de la face inférieure du cube mobile dans sa position de départ.

116) Saisie : mSS9 = repères2dsynchronisés[fSS9, fOr', fOx', fOy', mOr', mOx', mOy']

mSS10 sera le sommet de la section placé sur l’arête gauche de la face inférieure du cube mobile dans sa position de départ.

117) Saisie : mSS10 = repères2dsynchronisés[fSS10, fOr', fOx', fOy', mOr', mOx', mOy']

mSS11 sera le sommet de la section placé sur l’arête avant de la face inférieure du cube mobile dans sa position de départ.

118) Saisie : mSS11 = repères2dsynchronisés[fSS11, fOr', fOx', fOy', mOr', mOx', mOy']

mSS12 sera le sommet de la section placé sur l’arête droite de la face inférieure du cube mobile dans sa position de départ.

119) Saisie : mSS12 = repères2dsynchronisés[fSS12, fOr', fOx', fOy', mOr', mOx', mOy']

Pour tous ces points, dans Propriétés, onglet Avancé, Condition pour afficher l’objet, nous écrirons la condition suivante : sommetssection et nous n’afficherons pas les étiquettes.

Ainsi, ces sommets ne seront visibles que lorsque la case « sommets section » sera cochée.

<u>Déterminons les bords de la section dans le cube mobile</u>.

Voici un schéma montrant la construction de mco1.

```
Si mSS1 est défini
|   Alors  Si mSS2 est défini
|          |  Alors mco1 est le segment mSS1mSS2
|          |  Sinon  Si mSS3 est défini
|          |         |  Alors « mco1 » est le segment mSS1mSS3
|          |         |  Sinon   Si mSS4 est défini
|          |         |          | Alors mco1 est le segment mSS1mSS4
|          |         |          Fin si alors
|          |         Fin si alors sinon
|          Fin si alors sinon
|      Sinon Si     mSS2 est défini
|          |  Alors  Si mSS3 est défini
|          |         |  Alors mco1 est le segment mSS2mSS3
|          |         |  Sinon   Si mSS4 est défini
|          |         |          | Alors mco1 est le segment mSS2mSS4
|          |         |          Fin si alors
|          |         Fin si alors sinon
|          |  Sinon mco1 est le segment mSS3mSS4
|          Fin si alors sinon
Fin si alors sinon
```

120) Saisie : mco1 = Si[EstDéfini[mSS1], Si[EstDéfini[mSS2], Segment[mSS1, mSS2], Si[EstDéfini[mSS3], Segment[mSS1, mSS3], Si[EstDéfini[mSS4], Segment[mSS1, mSS4]]]], Si[EstDéfini[mSS2], Si[EstDéfini[mSS3], Segment[mSS2, mSS3], Si[EstDéfini[mSS4], Segment[mSS2, mSS4]]], Segment[mSS3, mSS4]]] (couleur : noir, épaisseur 4)

Dans Propriétés, onglet Avancé, Condition pour afficher l'objet, nous écrirons la condition suivante : (¬ppoints) ∧ bordssection ∧ indbssupérieure

Ainsi, « mco1 » ne sera visible que lorsque la case « bords de section » sera cochée et que la case « choisir les faces » sera décochée et que la valeur de l'indicateur indbsdroite sera true.

De manière analogue, on a

121) Saisie : mco1' = Si[EstDéfini[mSS9], Si[EstDéfini[mSS10], Segment[mSS9, mSS10], Si[EstDéfini[mSS11], Segment[mSS9, mSS11], Si[EstDéfini[mSS12], Segment[mSS9, mSS12]]]], Si[EstDéfini[mSS10], Si[EstDéfini[mSS11], Segment[mSS10, mSS11], Si[EstDéfini[mSS12], Segment[mSS10, mSS12]]], Segment[mSS11, mSS12]]] (couleur : noir, épaisseur 4, Propriétés, onglet Avancé, Condition pour afficher l'objet : (¬ppoints) ∧ bordssection ∧ indbsinférieure)

122) Saisie : mco2 = Si[EstDéfini[mSS4], Si[EstDéfini[mSS5], Segment[mSS4, mSS5], Si[EstDéfini[mSS8], Segment[mSS4, mSS8], Si[EstDéfini[mSS12], Segment[mSS4, mSS12]]]], Si[EstDéfini[mSS5], Si[EstDéfini[mSS8], Segment[mSS5, mSS8], Si[EstDéfini[mSS12], Segment[mSS5, mSS12]]], Segment[mSS8, mSS12]]] (couleur : noir, épaisseur 4, Propriétés, onglet Avancé, Condition pour afficher l'objet : (¬ppoints) ∧ bordssection ∧ indbsdroite)

123) Saisie : mco2’ = Si[EstDéfini[mSS2], Si[EstDéfini[mSS6], Segment[mSS2, mSS6], Si[EstDéfini[mSS7], Segment[mSS2, mSS7], Si[EstDéfini[mSS10], Segment[mSS2, mSS10]]]], Si[EstDéfini[mSS6], Si[EstDéfini[mSS7], Segment[mSS6, mSS7], Si[EstDéfini[mSS10], Segment[mSS6, mSS10]]], Segment[mSS7, mSS10]]] (couleur : noir, épaisseur 4, Propriétés, onglet Avancé, Condition pour afficher l’objet : (¬ppoints) ∧ bordssection ∧ indbsgauche)

124) Saisie : mco3 = Si[EstDéfini[mSS3], Si[EstDéfini[mSS7], Segment[mSS3, mSS7], Si[EstDéfini[mSS8], Segment[mSS3, mSS8], Si[EstDéfini[mSS11], Segment[mSS3, mSS11]]]], Si[EstDéfini[mSS7], Si[EstDéfini[mSS8], Segment[mSS7, mSS8], Si[EstDéfini[mSS11], Segment[mSS7, mSS11]]], Segment[mSS8, mSS11]]] (couleur : noir, épaisseur 4, Propriétés, onglet Avancé, Condition pour afficher l’objet : (¬ppoints) ∧ bordssection ∧ indbsavant)

125) Saisie : mco3’ = Si[EstDéfini[mSS1], Si[EstDéfini[mSS5], Segment[mSS1, mSS5], Si[EstDéfini[mSS6], Segment[mSS1, mSS6], Si[EstDéfini[mSS9], Segment[mSS1, mSS9]]]], Si[EstDéfini[mSS5], Si[EstDéfini[mSS6], Segment[mSS5, mSS6], Si[EstDéfini[mSS9], Segment[mSS5, mSS9]]], Segment[mSS6, mSS9]]] (couleur : noir, épaisseur 4, Propriétés, onglet Avancé, Condition pour afficher l’objet : (¬ppoints) ∧ bordssection ∧ indbsarrière)

Créons des bords pointillés pour la section.

126) Saisie : mco1p = Si[caché, mco1] (couleur : noir, épaisseur 5, Style : pointillés, Propriétés, onglet Avancé, Condition pour afficher l’objet : (¬ppoints) ∧ bordssection ∧ (¬indbssupérieure))

127) Saisie : mco1’p = Si[caché, mco1’] (couleur : noir, épaisseur 5, Style : pointillés, Propriétés, onglet Avancé, Condition pour afficher l’objet : (¬ppoints) ∧ bordssection ∧ (¬indbsinférieure))

128) Saisie : mco2p = Si[caché, mco2] (couleur : noir, épaisseur 5, Style : pointillés, Propriétés, onglet Avancé, Condition pour afficher l’objet : (¬ppoints) ∧ bordssection ∧ (¬indbsdroite))

129) Saisie : mco2’p = Si[caché, mco2’] (couleur : noir, épaisseur 5, Style : pointillés, Propriétés, onglet Avancé, Condition pour afficher l’objet : (¬ppoints) ∧ bordssection ∧ (¬indbsgauche))

130) Saisie : mco3p = Si[caché, mco3] (couleur : noir, épaisseur 5, Style : pointillés, Propriétés, onglet Avancé, Condition pour afficher l’objet : (¬ppoints) ∧ bordssection ∧ (¬indbsavant))

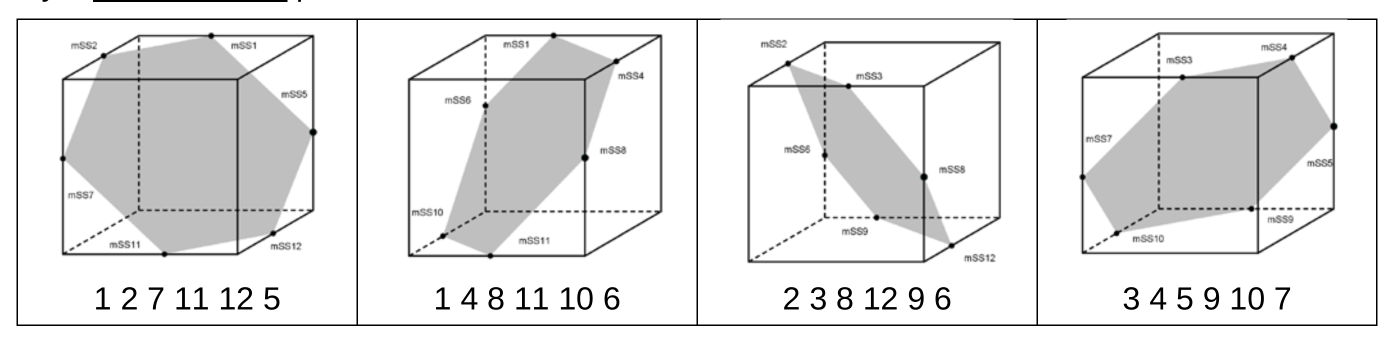
131) Saisie : mco3’p = Si[caché, mco3’] (couleur : noir, épaisseur 5, Style : pointillés, Propriétés, onglet Avancé, Condition pour afficher l’objet : (¬ppoints) ∧ bordssection ∧ (¬indbsarrière))

Déterminons la section non confondue avec une face.

Pour une écriture simplifiée, nous désignerons un polygone par la suite des numéros de ses sommets, par exemple 1 2 7 11 12 5 pour mSS1mSS2mSS7mSS11mSS12mSS5.

La section non particulière sera un *hexagone* lorsque le plan coupera chacune des six faces du cube, sera un *pentagone* lorsque le plan coupera cinq des six faces du cube, sera un *quadrilatère* lorsque le plan coupera quatre des six faces du cube et sera un triangle lorsque le plan coupera trois des six faces du cube.

Il y a ***4 hexagones*** possibles :



1 2 7 11 12 5 | 1 4 8 11 10 6 | 2 3 8 12 9 6 | 3 4 5 9 10 7

Un hexagone perdant 2 sommets consécutifs devient un pentagone.

Chaque hexagone peut perdre 2 sommets consécutifs de 5 manières.

1 2 7 11 12 5 perd 2 sommets consécutifs :
Perdre 1 2 n'est pas possible car il faudrait pour cela que fA et fB ne soient pas distincts,

S'il perd 2 7, il devient 1 3 11 12 5
S'il perd 7 11, il devient 1 2 10 12 5
S'il perd 11 12, il devient 1 2 7 8 5
S'il perd 12 5, il devient 1 2 7 11 9
S'il perd 1 5, il devient 2 4 12 11 7

1 4 8 11 10 6 perd 2 sommets consécutifs :
Perdre 1 4 n'est pas possible car il faudrait pour cela que fA et fB ne soient pas distincts,

S'il perd 1 6, il devient 2 4 8 11 10
S'il perd 4 8, il devient 1 3 11 10 6
S'il perd 8 11, il devient 1 4 12 10 6
S'il perd 11 10, il devient 1 4 8 7 6
S'il perd 10 6, il devient 1 4 8 11 9

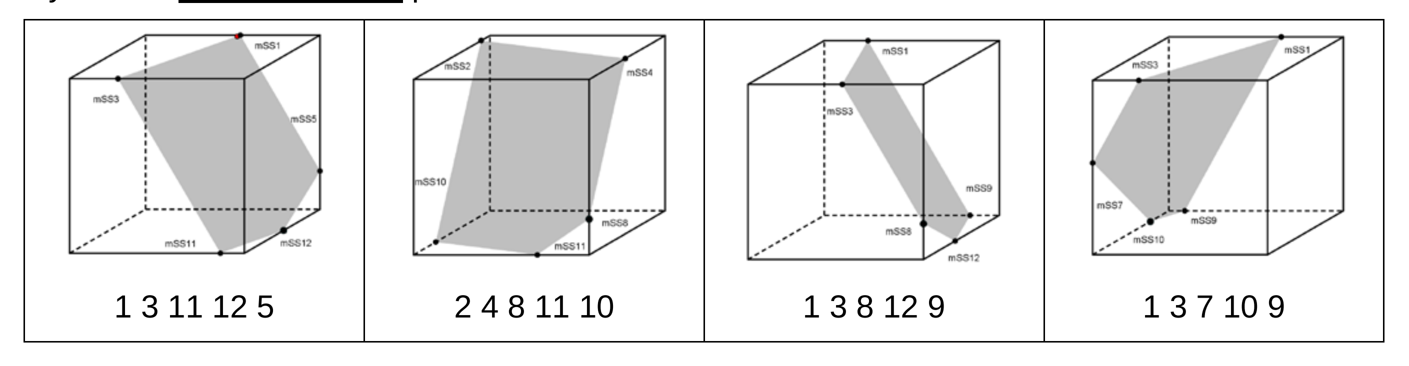
2 3 8 12 9 6 perd 2 sommets consécutifs :
Perdre 2 3 n'est pas possible car il faudrait pour cela que fA et fB ne soient pas distincts,

S'il perd 2 6, il devient 1 3 8 12 9
S'il perd 3 8, il devient 2 4 12 9 6
S'il perd 8 12, il devient 2 3 11 9 6
S'il perd 9 12, il devient 2 3 8 5 6
S'il perd 9 6, il devient 2 3 8 12 10

3 4 5 9 10 7 perd 2 sommets consécutifs :
Perdre 3 4 n'est pas possible car il faudrait pour cela que fA et fB ne soient pas distincts,

S'il perd 4 5, il devient 1 3 7 10 9
S'il perd 5 9, il devient 3 4 12 10 7
S'il perd 9 10, il devient 3 4 5 6 7
S'il perd 10 7, il devient 3 4 5 9 11
S'il perd 3 7, il devient 2 4 5 9 10

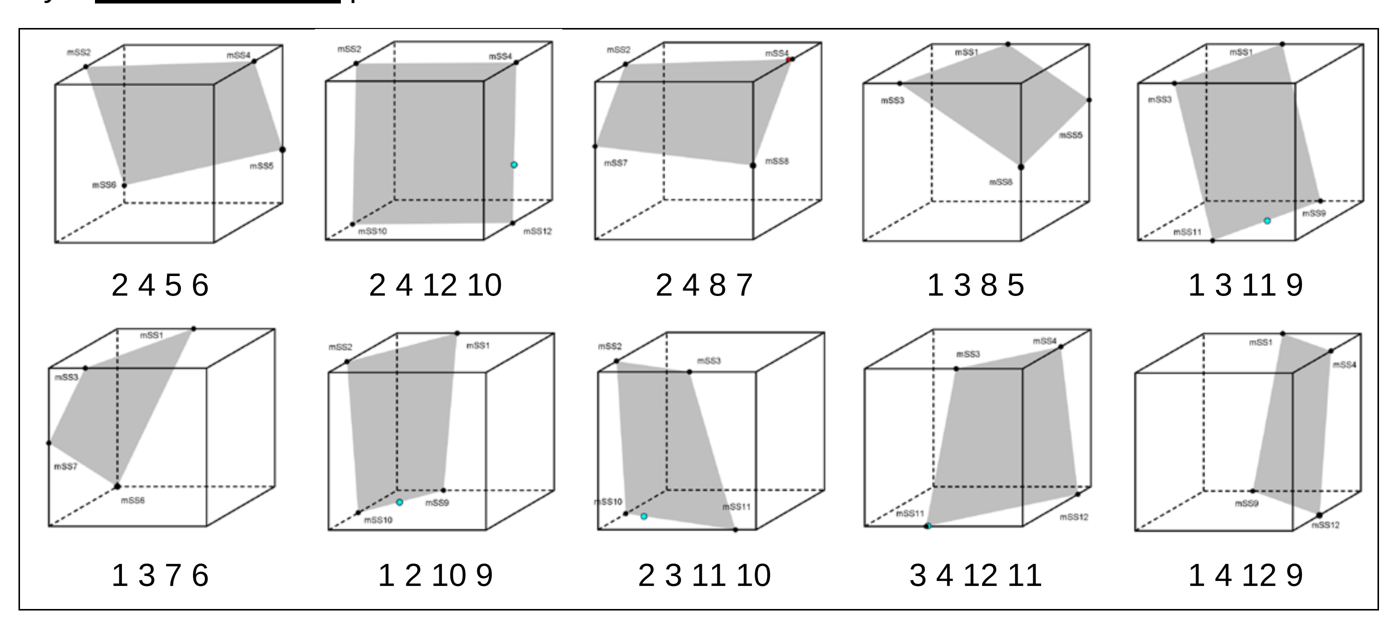
Il y a donc ***20 pentagones*** possibles :



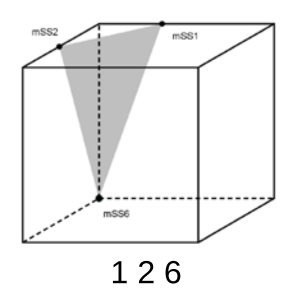
1 3 11 12 5 | 2 4 8 11 10 | 1 3 8 12 9 | 1 3 7 10 9

1 2 10 12 5	1 3 11 10 6	2 4 12 9 6	3 4 12 10 7
1 2 7 8 5	1 4 12 10 6	2 3 11 9 6	3 4 5 6 7
1 2 7 11 9	1 4 8 7 6	2 3 8 5 6	3 4 5 9 11
2 4 12 11 7	1 4 8 11 9	2 3 8 12 10	2 4 5 9 10

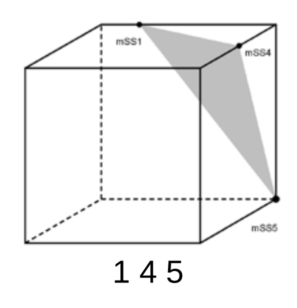
Il y a ***10 quadrilatères*** possibles :



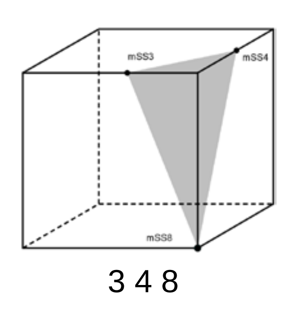
Il y a ***4 triangles*** possibles :



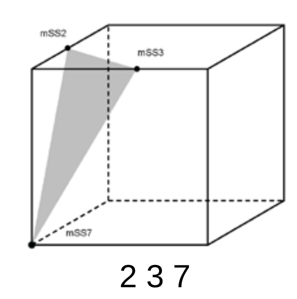
1 2 6



1 4 5



3 4 8



2 3 7

Il y a donc 38 polygones possibles pour déterminer la section.

Pour établir une seule formule Geogebra donnant cette section, nous trierons ces 38 cas selon les numéros des sommets des polygones, en ordre croissant, de gauche à droite (cela se réalise rapidement dans un tableau Excel), puis nous raisonnerons comme le montrent les schémas suivants.

1	2	6			
1	2	7	8	5	
1	2	7	11	9	
1	2	7	11	12	5
1	2	10	9		
1	2	10	12	5	
1	3	7	6		
1	3	7	10	9	
1	3	8	5		
1	3	8	12	9	
1	3	11	9		
1	3	11	10	6	
1	3	11	12	5	
1	4	5			
1	4	8	7	6	
1	4	8	11	9	
1	4	8	11	10	6
1	4	12	9		
1	4	12	10	6	
2	3	7			
2	3	8	5	6	
2	3	8	12	9	6
2	3	8	12	10	
2	3	11	9	6	
2	3	11	10		
2	4	5	6		
2	4	5	9	10	
2	4	8	7		
2	4	8	11	10	
2	4	12	9	6	
2	4	12	10		
2	4	12	11	7	
3	4	5	6	7	
3	4	5	9	10	7
3	4	5	9	11	
3	4	8			
3	4	12	10	7	
3	4	12	11		

Gérons la colonne 1

Si mSS1 existe
| alors gérer la suite pour mSS1 en colonne 1
| Sinon si mSS2 existe
| | Alors gérer la suite pour mSS2 en colonne 1
| | Sinon gérer la suite pour mSS3 en colonne 1
| Fin si alors sinon
Fin si alors sinon

Pour Geogebra, on a donc la formule provisoire

section = Si[EstDéfini[mSS1], gérer la suite pour mSS1 en colonne 1, Si[EstDéfini[mSS2], gérer la suite pour mSS2 en colonne 1, gérer la suite pour mSS3 en colonne 1]]

1	2	6			
1	2	7	8	5	
1	2	7	11	9	
1	2	7	11	12	5
1	2	10	9		
1	2	10	12	5	
1	3	7	6		
1	3	7	10	9	
1	3	8	5		
1	3	8	12	9	
1	3	11	9		
1	3	11	10	6	
1	3	11	12	5	
1	4	5			
1	4	8	7	6	
1	4	8	11	9	
1	4	8	11	10	6
1	4	12	9		
1	4	12	10	6	

Gérons la suite pour mSS1 en colonne 1

Si mSS2 existe
| alors gérer la suite pour mSS2 en colonne2
| Sinon si mSS3 existe
| | Alors gérer la suite pour mSS3 en colonne 2
| | Sinon gérer la suite pour mSS4 en colonne 2
| Fin si alors sinon
Fin si alors sinon

En Geogebra, cela donne

Si[EstDéfini[mSS2], gérer la suite pour mSS2 en colonne 2, Si[EstDéfini[mSS3], gérer la suite pour mSS3 en colonne 2, gérer la suite pour mSS4 en colonne 2]]

Gérons la suite pour mSS2 en colonne 1Si mSS3 existe
| alors gérer la suite pour mSS3 en colonne 2
| sinon gérer la suite pour mSS4 en colonne 2
Fin si alors sinon

2	3	7			
2	3	8	5	6	
2	3	8	12	9	6
2	3	8	12	10	
2	3	11	9	6	
2	3	11	10		
2	4	5	6		
2	4	5	9	10	
2	4	8	11	10	
2	4	12	9	6	
2	4	12	10		
2	4	12	11	7	

En Geogebra, cela donne

Si[EstDéfini[mSS3], gérer la suite pour mSS3 en colonne 2, gérer la suite pour mSS4 en colonne 2]

Gérons la suite pour mSS3 en colonne 1

Si mSS5 existe
| Alors gérer la suite pour mSS5 en colonne 3
| Sinon si mSS8 existe
| | Alors section = 3 4 8
| | Sinon gérer la suite pour mSS12 en colonne 3
| Fin si alors sinon
Fin si alors sinon

3	4	5	6	7	
3	4	5	9	10	7
3	4	5	9	11	
3	4	8			
3	4	12	10	7	
3	4	12	11		

En Geogebra, cela donne

Si[EstDéfini[mSS5], gérer la suite pour mSS5 en colonne 3, Si[EstDéfini[mSS8], Polygone[mSS3,mSS4,mSS8], gérer la suite pour mSS12 en colonne 3]]

Pour Geogebra, la formule provisoire devient

section = Si[EstDéfini[mSS1], Si[EstDéfini[mSS2], gérer la suite pour mSS2 en colonne 2, Si[EstDéfini[mSS3], gérer la suite pour mSS3 en colonne 2,gérer la suite pour mSS4 en colonne 2]], Si[EstDéfini[mSS2], Si[EstDéfini[mSS3], gérer la suite pour mSS3 en colonne 2, gérer la suite pour mSS4 en colonne 2], Si[EstDéfini[mSS5], gérer la suite pour mSS5 en colonne 3, Si[EstDéfini[mSS8], Polygone[mSS3,mSS4,mSS8], gérer la suite pour mSS12 en colonne 3]]]]

Avec mSS1 en colonne 1, gérons la suite pour mSS2 en colonne 2

Si mSS6 existe
| alors section = 1 2 6
| Sinon si mSS7 existe
| | Alors gérer la suite pour mSS7 en colonne 3
| | Sinon gérer la suite pour mSS10 en colonne 3
| Fin si alors sinon
Fin si alors sinon

1	2	6			
1	2	7	8	5	
1	2	7	11	9	
1	2	7	11	12	5
1	2	10	9		
1	2	10	12	5	

En Geogebra, cela donne

Si[EstDéfini[mSS6], Polygone[mSS1, mSS2, mSS6], Si[EstDéfini[mSS7], gérer la suite pour mSS7 en colonne 3, gérer la suite pour mSS10 en colonne 3]]

Avec mSS1 en colonne 1, gérons la suite pour mSS3 en colonne 2

Si mSS7 existe
| alors gérer la suite pour mSS7 en colonne 3
| Sinon si mSS8 existe
| | Alors gérer la suite pour mSS8 en colonne 3
| | Sinon gérer la suite pour mSS11 en colonne 3
| Fin si alors sinon
Fin si alors sinon

1	3	7	6	
1	3	7	10	9
1	3	8	5	
1	3	8	12	9
1	3	11	9	
1	3	11	10	6
1	3	11	12	5

En Geogebra, cela donne

Si[EstDéfini[mSS7], gérer la suite pour mSS7 en colonne 3, Si[EstDéfini[mSS8], gérer la suite pour mSS8 en colonne 3, gérer la suite pour mSS11 en colonne 3]]

Avec mSS1 en colonne 1, gérons la suite pour mSS4 en colonne 2

Si mSS5 existe
| alors section = 1 4 5
| Sinon si mSS8 existe
| | Alors gérer la suite pour mSS8 en colonne 3
| | Sinon gérer la suite pour mSS12 en colonne 3
| Fin si alors sinon
Fin si alors sinon

1	4	5			
1	4	8	7	6	
1	4	8	11	9	
1	4	8	11	10	6
1	4	12	9		
1	4	12	10	6	

En Geogebra, cela donne

Si[EstDéfini[mSS5], Polygone[mSS1,mSS4,mSS5], Si[EstDéfini[mSS8], gérer la suite pour mSS8 en colonne 3, gérer la suite pour mSS12 en colonne 3]]

Avec mSS2 en colonne 1, gérons la suite pour mSS3 en colonne 2

Si mSS7 existe
| alors section = 2 3 7
| Sinon si mSS8 existe
| | Alors section = gérer la suite pour mSS8 en colonne 3
| | Sinon section = gérer la suite pour mSS11 en colonne 3
| Fin si alors sinon
Fin si alors sinon

2	3	7			
2	3	8	5	6	
2	3	8	12	9	6
2	3	8	12	10	
2	3	11	9	6	
2	3	11	10		

En Geogebra, cela donne

Si[EstDéfini[mSS7], Polygone[mSS2,mSS3,mSS7], Si[EstDéfini[mSS8], gérer la suite pour mSS8 en colonne 3, gérer la suite pour mSS11 en colonne 3]]

Avec mSS2 en colonne 1, gérons la suite pour mSS4 en colonne 2

Si mSS5 existe
| alors section = gérer la suite pour mSS5 en colonne3
| Sinon si mSS8 existe
| | Alors section = gérer la suite pour mSS8 en colonne3
| | Sinon section = gérer la suite pour mSS12 en colonne3
| Fin si alors sinon
Fin si alors sinon

2	4	5	6	
2	4	5	9	10
2	4	8	7	
2	4	8	11	10
2	4	12	9	6
2	4	12	10	
2	4	12	11	7

En Geogebra, cela donne

Si[EstDéfini[mSS5], gérer la suite pour mSS5 en colonne3, Si[EstDéfini[mSS8], gérer la suite pour mSS8 en colonne3, gérer la suite pour mSS12 en colonne3]]

Avec mSS3 en colonne 1, gérons la suite pour mSS5 en colonne 3

Si mSS6 existe
| alors section = 3 4 5 6 7
| alors gérer la suite pour mSS9 en colonne 4
Fin si alors sinon

3	4	5	6	7	
3	4	5	9	10	7
3	4	5	9	11	

En Geogebra, cela donne

Si[EstDéfini[mSS6], Polygone[mSS3,mSS4,mSS5,mSS6,mSS7], gérer la suite pour mSS9 en colonne 4]

Avec mSS3 en colonne 1, gérons la suite pour mSS12 en colonne 3

Si mSS10 existe
| Alors section = 3 4 12 10 7
| Sinon section = 3 4 12 11
Fin si alors sinon

3	4	12	10	7
3	4	12	11	

En Geogebra, cela donne

Si[EstDéfini[mSS10], Polygone[mSS3,mSS4,mSS12, mSS10, mSS7], Polygone[mSS3,mSS4,mSS12, mSS11]]

Pour Geogebra, la formule provisoire devient

section = Si[EstDéfini[mSS1], Si[EstDéfini[mSS2], Si[EstDéfini[mSS6], Polygone[mSS1, mSS2, mSS6], Si[EstDéfini[mSS7], gérer la suite pour mSS7 en colonne 3, gérer la suite pour mSS10 en colonne 3]], Si[EstDéfini[mSS3], Si[EstDéfini[mSS7], gérer la suite pour mSS7 en colonne 3, Si[EstDéfini[mSS8], gérer la suite pour mSS8 en colonne 3, gérer la suite pour mSS11 en colonne 3]], Si[EstDéfini[mSS5], Polygone[mSS1,mSS4,mSS5], Si[EstDéfini[mSS8], gérer la suite pour mSS8 en colonne 3, gérer la suite pour mSS12 en colonne 3]]]], Si[EstDéfini[mSS2], Si[EstDéfini[mSS3], Si[EstDéfini[mSS7], Polygone[mSS2,mSS3,mSS7], Si[EstDéfini[mSS8], gérer la suite pour mSS8 en colonne 3, gérer la suite pour mSS11 en colonne 3]], Si[EstDéfini[mSS5], gérer la suite pour mSS5 en colonne3, Si[EstDéfini[mSS8], gérer la suite pour mSS8 en colonne3, gérer la suite pour mSS12 en colonne3]]], Si[EstDéfini[mSS5], Si[EstDéfini[mSS6], Polygone[mSS3,mSS4,mSS5,mSS6,mSS7], gérer la suite pour mSS9 en colonne 4], Si[EstDéfini[mSS8], Polygone[mSS3,mSS4,mSS8], Si[EstDéfini[mSS10], Polygone[mSS3,mSS4,mSS12, mSS10, mSS7], Polygone[mSS3,mSS4,mSS12, mSS11]]]]]]

Avec mSS1 en colonne 1 et mSS2 en colonne 2, gérons la suite pour mSS7 en colonne 3

Si mSS8 existe
| Alors section = 1 2 7 8 5
| Sinon si mSS9 existe
| | alors section = 1 2 7 11 9
| | sinon section = 1 2 7 11 12 5
| Fin si alors sinon
Fin si alors sinon

1	2	7	8	5	
1	2	7	11	9	
1	2	7	11	12	5

En Geogebra, cela donne

Si[EstDéfini[mSS8], Polygone[mSS1,mSS2,mSS7,mSS8, mSS5], Si[EstDéfini[mSS9], Polygone[mSS1,mSS2,mSS7,mSS11,mSS9], Polygone[mSS1,mSS2,mSS7,mSS11,mSS12, mSS5]]]

Avec mSS1 en colonne1 et mSS2 en colonne 2, gérons la suite pour mSS10 en colonne3

Si mSS9 existe
| alors section = 1 2 10 9
| alors section = 1 2 10 12 5
Fin si alors sinon

1	2	10	9	
1	2	10	12	5

En Geogebra, cela donne

Si[EstDéfini[mSS9], Polygone[mSS1,mSS2,mSS10,mSS9], Polygone[mSS1,mSS2,mSS10,mSS12,mSS5]]

Avec mSS1 en colonne1 et mSS3 en colonne 2, gérons la suite pour mSS7 en colonne3

Si mSS6 existe
| alors section = 1 3 7 6
| alors section = section = 1 3 7 10 9
Fin si alors sinon

1	3	7	6	
1	3	7	10	9

En Geogebra, cela donne

Si[EstDéfini[mSS6], Polygone[mSS1,mSS3,mSS7,mSS6], Polygone[mSS1,mSS3,mSS7,mSS10, mSS9]]

Avec mSS1 en colonne1 et mSS3 en colonne 2, gérons la suite pour mSS8 en colonne3

Si mSS5 existe
| alors section = 1 3 8 5
| alors section = section = 1 3 8 12 9
Fin si alors sinon

1	3	8	5	
1	3	8	12	9

En Geogebra, cela donne

Si[EstDéfini[mSS5], Polygone[mSS1,mSS3,mSS8,mSS5], Polygone[mSS1,mSS3,mSS8,mSS12, mSS9]]

Avec mSS1 en colonne1 et mSS3 en colonne 2, gérons la suite pour mSS8 en colonne3

Si mSS9 existe
| Alors section = 1 3 11 9
| Sinon si mSS10 existe
| | alors section = 1 3 11 10 6
| | sinon section = 1 3 11 12 5
| Fin si alors sinon
Fin si alors sinon

1	3	11	9	
1	3	11	10	6
1	3	11	12	5

En Geogebra, cela donne

Si[EstDéfini[mSS9], Polygone[mSS1,mSS3,mSS11,mSS9], Si[EstDéfini[mSS10], Polygone[mSS1,mSS3,mSS11,mSS10,mSS6], Polygone[mSS1,mSS3,mSS11,mSS12,mSS5]]]

Avec mSS1 en colonne1 et mSS4 en colonne 2, gérons la suite pour mSS8 en colonne3

Si mSS7 existe
| Alors section = 1 4 8 7 6
| Sinon si mSS9 existe
| | alors section = 1 4 8 11 9
| | sinon section = 1 4 8 11 10 6
| Fin si alors sinon
Fin si alors sinon

1	4	8	7	6	
1	4	8	11	9	
1	4	8	11	10	6

En Geogebra, cela donne

Si[EstDéfini[mSS7], Polygone[mSS1,mSS4,mSS8,mSS7, mSS6], Si[EstDéfini[mSS9], Polygone[mSS1,mSS4,mSS8,mSS11,mSS9], Polygone[mSS1,mSS4,mSS8,mSS11,mSS10, mSS6]]]

Avec mSS1 en colonne1 et mSS3 en colonne 2, gérons la suite pour mSS8 en colonne3

Si mSS9 existe
| alors section = 1 4 12 9
| alors section = section = 1 4 12 10 6
Fin si alors sinon

1	4	12	9	
1	4	12	10	6

En Geogebra, cela donne

Si[EstDéfini[mSS9], Polygone[mSS1,mSS4,mSS12,mSS9], Polygone[mSS1,mSS4,mSS12,mSS10, mSS6]]

Avec mSS2 en colonne1 et mSS3 en colonne 2, gérons la suite pour mSS8 en colonne3

Si mSS5 existe
| Alors section = 2 3 8 5 6
| Sinon si mSS9 existe
| | alors section = 2 3 8 12 9 6
| | sinon section = 2 3 8 12 10
| Fin si alors sinon
Fin si alors sinon

2 3	8	5	6		
2 3	8	12		9	6
2 3	8	12	10		

En Geogebra, cela donne

Si[EstDéfini[mSS5], Polygone[mSS2,mSS3,mSS8,mSS5,mSS6], Si[EstDéfini[mSS9], Polygone[mSS2,mSS3,mSS8,mSS12,mSS9,mSS6], Polygone[mSS2,mSS3,mSS8,mSS12,mSS10]]]

Avec mSS2 en colonne1 et mSS3 en colonne 2, gérons la suite pour mSS8 en colonne3

Si mSS9 existe
| alors section = 2 3 11 9 6
| alors section = section = 2 3 11 10
Fin si alors sinon

2 3	11	9	6
2 3	11	10	

En Geogebra, cela donne

Si[EstDéfini[mSS9], Polygone[mSS2,mSS3,mSS11,mSS9,mSS6], Polygone[mSS2,mSS3,mSS11,mSS10]]

Avec mSS2 en colonne1 et mSS4 en colonne 2, gérons la suite pour mSS5 en colonne3

Si mSS6 existe
| alors section = 2 4 5 6
| alors section = section = 2 4 5 9 10
Fin si alors sinon

2 4	5	6	
2 4	5	9	10

En Geogebra, cela donne

Si[EstDéfini[mSS6], Polygone[mSS2,mSS4,mSS5,mSS6], Polygone[mSS2,mSS4,mSS5,mSS9,mSS10]]

Avec mSS2 en colonne1 et mSS4 en colonne 2, gérons la suite pour mSS8 en colonne3

Si mSS7 existe
| alors section = 2 4 8 7
| alors section = section = 2 4 8 11 10
Fin si alors sinon

2 4	8	7	
2 4	8	11	10

En Geogebra, cela donne

Si[EstDéfini[mSS7], Polygone[mSS2,mSS4,mSS8,mSS7], Polygone[mSS2,mSS4,mSS8,mSS11,mSS10]]

Avec mSS2 en colonne1 et mSS4 en colonne 2, gérons la suite pour mSS12 en colonne3

Si mSS9 existe
| Alors section = 2 4 12 9 6
| Sinon si mSS10 existe
| | alors section = 2 4 12 10
| | sinon section = 2 4 12 11 7
| Fin si alors sinon
Fin si alors sinon

2 4	12	9	6
2 4	12	10	
2 4	12	11	7

En Geogebra, cela donne

Si[EstDéfini[mSS9], Polygone[mSS2,mSS4,mSS12,mSS9, mSS6], Si[EstDéfini[mSS10], Polygone[mSS2,mSS4,mSS12,mSS10], Polygone[mSS2,mSS4,mSS12,mSS11,mSS7]]]
Avec mSS3 en colonne1 et mSS5 en colonne 3, gérons la suite pour mSS9 en colonne 4

3	4	5	9	10	7
3	4	5	9	11	

Si mSS10 existe
| alors section = 3 4 5 9 10 7
| alors section = 3 4 5 9 11
Fin si alors sinon

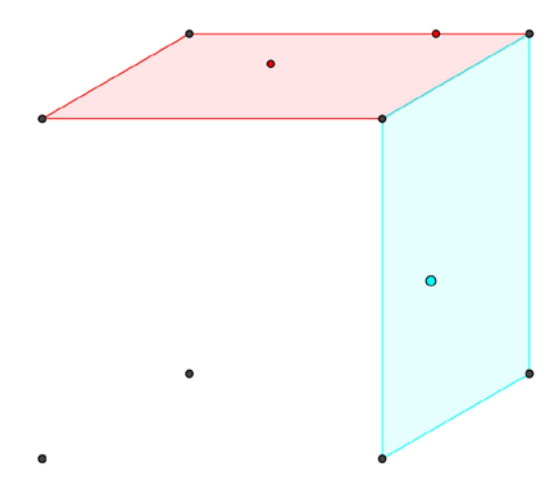
En Geogebra, cela donne

Si[EstDéfini[mSS10], Polygone[mSS3,mSS4,mSS5,mSS9,mSS10,mSS7], Polygone[mSS3,mSS4,mSS5,mSS9,mSS11]]

Pour Geogebra, la formule devient finalement

132) Saisie : section = Si[EstDéfini[mSS1], Si[EstDéfini[mSS2], Si[EstDéfini[mSS6], Polygone[mSS1, mSS2, mSS6], Si[EstDéfini[mSS7], Si[EstDéfini[mSS8], Polygone[mSS1, mSS2, mSS7, mSS8, mSS5], Si[EstDéfini[mSS9], Polygone[mSS1, mSS2, mSS7, mSS11, mSS9], Polygone[mSS1, mSS2, mSS7, mSS11, mSS12, mSS5]]], Si[EstDéfini[mSS9], Polygone[mSS1, mSS2, mSS10, mSS9], Polygone[mSS1, mSS2, mSS10, mSS12, mSS5]]]], Si[EstDéfini[mSS3], Si[EstDéfini[mSS7], Si[EstDéfini[mSS6], Polygone[mSS1, mSS3, mSS7, mSS6], Polygone[mSS1, mSS3, mSS7, mSS10, mSS9]], Si[EstDéfini[mSS8], Si[EstDéfini[mSS5], Polygone[mSS1, mSS3, mSS8, mSS5], Polygone[mSS1, mSS3, mSS8, mSS12, mSS9]], Si[EstDéfini[mSS9], Polygone[mSS1, mSS3, mSS11, mSS9], Si[EstDéfini[mSS10], Polygone[mSS1, mSS3, mSS11, mSS10, mSS6], Polygone[mSS1, mSS3, mSS11, mSS12, mSS5]]]]], Si[EstDéfini[mSS5], Polygone[mSS1, mSS4, mSS5], Si[EstDéfini[mSS8], Si[EstDéfini[mSS7], Polygone[mSS1, mSS4, mSS8, mSS7, mSS6], Si[EstDéfini[mSS9], Polygone[mSS1, mSS4, mSS8, mSS11, mSS9], Polygone[mSS1, mSS4, mSS8, mSS11, mSS10, mSS6]]], Si[EstDéfini[mSS9], Polygone[mSS1, mSS4, mSS12, mSS9], Polygone[mSS1, mSS4, mSS12, mSS10, mSS6]]]]]], Si[EstDéfini[mSS2], Si[EstDéfini[mSS3], Si[EstDéfini[mSS7], Polygone[mSS2, mSS3, mSS7], Si[EstDéfini[mSS8], Si[EstDéfini[mSS5], Polygone[mSS2, mSS3, mSS8, mSS5, mSS6], Si[EstDéfini[mSS9], Polygone[mSS2, mSS3, mSS8, mSS12, mSS9, mSS6], Polygone[mSS2, mSS3, mSS8, mSS12, mSS10]]], Si[EstDéfini[mSS9], Polygone[mSS2, mSS3, mSS11, mSS9, mSS6], Polygone[mSS2, mSS3, mSS11, mSS10]]]], Si[EstDéfini[mSS5], Si[EstDéfini[mSS6], Polygone[mSS2, mSS4, mSS5, mSS6], Polygone[mSS2, mSS4, mSS5, mSS9, mSS10]], Si[EstDéfini[mSS8], Si[EstDéfini[mSS7], Polygone[mSS2, mSS4, mSS8, mSS7], Polygone[mSS2, mSS4, mSS8, mSS11, mSS10]], Si[EstDéfini[mSS9], Polygone[mSS2, mSS4, mSS12, mSS9, mSS6], Si[EstDéfini[mSS10], Polygone[mSS2, mSS4, mSS12, mSS10], Polygone[mSS2, mSS4, mSS12, mSS11, mSS7]]]]]], Si[EstDéfini[mSS5], Si[EstDéfini[mSS6], Polygone[mSS3, mSS4, mSS5, mSS6, mSS7], Si[EstDéfini[mSS10], Polygone[mSS3, mSS4, mSS5, mSS9, mSS10, mSS7], Polygone[mSS3, mSS4, mSS5, mSS9, mSS11]]], Si[EstDéfini[mSS8], Polygone[mSS3, mSS4, mSS8], Si[EstDéfini[mSS10], Polygone[mSS3, mSS4, mSS12, mSS10, mSS7], Polygone[mSS3, mSS4, mSS12, mSS11]]]]]] (couleur : noir, épaisseur du trait : 0, opacité : 25, Condition pour afficher l'objet : vuesection)

Pour choisir les faces.



Pour visualiser les faces sur lesquelles nous poserons « fA », « fB » et « fC », nous créerons deux polygones visibles seulement lorsque la boîte à cocher « choisir les faces » (ppoints) sera cochée.

133) Faab = Si[tp1 ≟ 1, Polygone[fS4, fS1, fS2, fS3], Si[tp1 ≟ 2, Polygone[fS4, fS1, fS5, fS8], Si[tp1 ≟ 3, Polygone[fS1, fS2, fS6, fS5], Si[tp1 ≟ 4, Polygone[fS2, fS3, fS7, fS6], Si[tp1 ≟ 5, Polygone[fS4, fS3, fS7, fS8], Polygone[fS5, fS6, fS7, fS8]]]]]] (couleur : rouge, opacité : 10, Condition pour afficher l'objet : ppoints)

134) Fac = Si[tp2 ≟ 1, Polygone[fS4, fS1, fS2, fS3], Si[tp2 ≟ 2, Polygone[fS4, fS1, fS5, fS8], Si[tp2 ≟ 3, Polygone[fS1, fS2, fS6, fS5], Si[tp2 ≟ 4, Polygone[fS2, fS3, fS7, fS6], Si[tp2 ≟ 5, Polygone[fS4, fS3, fS7, fS8], Polygone[fS5, fS6, fS7, fS8]]]]]] (couleur : bleu, opacité : 10, Condition pour afficher l'objet : ppoints)

Réglons l'affichage de « fA », « fB » et « fC ».

Pour chacun de ces points, dans Propriétés, Conditions pour afficher l'objet, écrivons

$\alpha \stackrel{?}{=} 0° \wedge \beta \stackrel{?}{=} 0° \wedge \gamma \stackrel{?}{=} 0°$.

« fA », « fB » et « fC » ne s'afficherons donc que lorsque le cube mobile sera dans sa position de départ. On pourra donc, sur un cube au repos, déplacer les points déterminant la section pour observer les transformations de celle-ci.

Réglons l'affichage de « mA », « mB » et « mC ».

Pour chacun de ces points, dans Propriétés, Conditions pour afficher l'objet, écrivons

$\neg(\alpha \stackrel{?}{=} 0° \wedge \beta \stackrel{?}{=} 0° \wedge \gamma \stackrel{?}{=} 0°)$.

« mA », « mB » et « mC » ne s'afficherons donc que lorsque le cube mobile ne sera pas dans sa position de départ.

Sélectionnables ou non ?

Dans les Propriétés de tous les objets, onglet Avancé, à l'exception de « fA », « fB » et « fC », des curseurs de paramètres et des boîtes à cocher, nous décocherons l'option sélectionnable. De cette manière, le passage de la souris n'accrochera que les points, curseurs ou boîte à cocher qui doivent l'être.

Et voilà. C'est fini !

Bonne coupe !